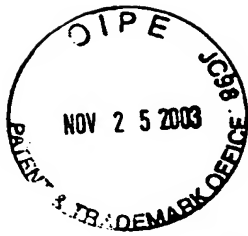


45982



PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of :  
Sang-Hyuck Ha et al. :  
Serial No.: 10/695,390 : Group Art Unit: Unassigned  
Filed: October 29, 2003 :  
For: METHOD AND APPARATUS FOR :  
DEINTERLEAVING INTERLEAVED :  
DATA STREAM IN A COMMUNICATION :  
SYSTEM :

TRANSMITTAL OF PRIORITY DOCUMENTS

COMMISSIONER FOR PATENTS  
P. O. Box 1450  
Alexandria, VA 22313-1450

Sir:

In order to perfect the claim for priority under 35 U.S.C. §119(a), the Applicants herewith submit certified copies of Korean Patent Application Nos. 2002-0066189, as filed on October 29, 2003 and 2003-0066400, as filed on September 24, 2003. Should anything further be required, the Office is asked to contact the undersigned attorney at the local telephone number listed below.

Respectfully submitted,

Peter L. Kendall  
Attorney of Record  
Reg. No.: 46,246

Roylance, Abrams, Berdo & Goodman, L.L.P.  
1300 19<sup>th</sup> Street, N.W., Suite 600  
Washington, D.C. 20036-2680  
(202) 659-9076

Dated: November 25, 2003



별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto  
is a true copy from the records of the Korean Intellectual  
Property Office.

출원번호 : 10-2002-0066189  
Application Number

출원년월일 : 2002년 10월 29일  
Date of Application OCT 29, 2002

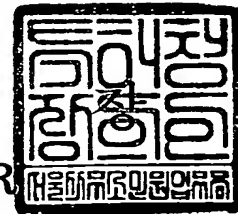
출원인 : 삼성전자주식회사  
Applicant(s) SAMSUNG ELECTRONICS CO., LTD.



2003 년 10 월 29 일

특 허 청

COMMISSIONER



## 【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0002
【제출일자】	2002.10.29
【국제특허분류】	H04B
【국제특허분류】	G06F
【발명의 명칭】	인터리빙된 데이터 열의 디인터리빙 방법 및 장치
【발명의 영문명칭】	METHOD AND APPARATUS FOR DEINTERLEAVING AN INTERLEAVED DATA STREAM IN COMMUNICATION SYSTEM
【출원인】	
【명칭】	삼성전자 주식회사
【출원인코드】	1-1998-104271-3
【대리인】	
【성명】	이건주
【대리인코드】	9-1998-000339-8
【포괄위임등록번호】	1999-006038-0
【발명자】	
【성명의 국문표기】	하상혁
【성명의 영문표기】	HA,SANG HYUCK
【주민등록번호】	730219-1167429
【우편번호】	441-390
【주소】	경기도 수원시 권선구 권선동 1314번지 주공1단지아파트 121동 1003 호
【국적】	KR
【발명자】	
【성명의 국문표기】	허서원
【성명의 영문표기】	HEO,Seo Weon
【주민등록번호】	670302-1670811
【우편번호】	152-099
【주소】	서울특별시 구로구 개봉본동 127-18
【국적】	KR

**【발명자】**

**【성명의 국문표기】** 유남열  
**【성명의 영문표기】** YU,Nam Yul  
**【주민등록번호】** 711207-1024118  
**【우편번호】** 442-725  
**【주소】** 경기도 수원시 팔달구 영통동 벽적골8단지 아파트 812-804  
**【국적】** KR

**【발명자】**

**【성명의 국문표기】** 김민구  
**【성명의 영문표기】** KIM,MIN Goo  
**【주민등록번호】** 640820-1067025  
**【우편번호】** 442-811  
**【주소】** 경기도 수원시 팔달구 영통동 968 신나무실 신명아파트 633-1502  
**【국적】** KR

**【발명자】**

**【성명의 국문표기】** 안성우  
**【성명의 영문표기】** AHN,Seong Woo  
**【주민등록번호】** 710128-1036821  
**【우편번호】** 449-741  
**【주소】** 경기도 용인시 기흥읍 신일아파트 202동 1101호  
**【국적】** KR

**【취지】**

특허법 제42조의 규정에 의하여 위와 같이 출원합니다. 대리인 이견주 (인)

**【수수료】**

<b>【기본출원료】</b>	20	면	29,000	원
<b>【가산출원료】</b>	40	면	40,000	원
<b>【우선권주장료】</b>	0	건	0	원
<b>【심사청구료】</b>	0	항	0	원
<b>【합계】</b>	69,000	원		

## 【요약서】

## 【요약】

본 발명은 고속 패킷 데이터 통신 시스템에서 인터리빙되어 전송된 데이터를 수신하여 원래의 형태로 디인터리빙하는 방법 및 장치에 관한 것으로서, 인터리빙된 순서대로 버퍼에 저장된 한 서브블럭의 부호심볼들을 디인터리빙된 순서대로 읽어내어 복호기로 입력하기 위한 방법에 대한 것이다. 여기서 상기 부호심볼들은 상기 서브블럭의 크기에 따라 미리 정해지는 개수의 행과 열을 가지는 행렬의 형태로 저장되어 있다. 임시주소 생성기는 상기 행렬의 마지막 열을 고려하지 않은 채 상기 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 임시주소를 생성하며, 주소 보상기는 상기 행렬의 마지막 열을 고려하여 상기 임시주소를 보상하기 위한 주소 보상값을 계산한다. 그러면 가산기는 상기 임시주소에 상기 주소 보상값을 더하여 상기 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 읽기주소를 생성한다. 이로써 본 발명은 필요한 메모리와 하드웨어 소자의 개수를 최소화하면서 인터리빙된 데이터의 열을 고속으로 디인터리빙함으로써 수신기의 소형화, 저전력화를 가능하게 한다.

## 【대표도】

도 5

## 【색인어】

1xEV-DV, deinterleaving, subblock deinterleaving, BRO

## 【명세서】

## 【발명의 명칭】

인터리빙된 데이터 열의 디인터리빙 방법 및 장치{METHOD AND APPARATUS FOR DEINTERLEAVING AN INTERLEAVED DATA STREAM IN COMMUNICATION SYSTEM}

## 【도면의 간단한 설명】

도 1은 순방향 패킷 데이터 채널을 위한 패킷 데이터를 인터리빙하여 송신 서브패킷을 생성하는 기지국 송신기의 구성도.

도 2는 채널 인터리버(120)에 의한 서브블럭 인터리빙과 서브블럭 심볼 그룹화 동작을 설명하는 도면.

도 3은 순방향 패킷 데이터 채널을 통해 수신된 서브패킷으로부터 복호 데이터를 얻는 단말 수신기의 구성도.

도 4는  $N_{EP}=408$ 인 경우 하나의 서브블럭에 대해 수행되는 인터리빙 과정을 세 단계로 분리하여 나타낸 도면.

도 3은 <실시예 1>에 제시된 인코더 패킷의 크기와 interleaving 파라미터를 적용한 경우, 부호심볼 인덱스, 행 인덱스, 임시 주소, 주소 보상분, 그리고 최종 읽기주소를 차례로 나타내어 도시하는 도면.

도 4는 <실시예 2>에 제시된 인코더 패킷의 크기와 interleaving 파라미터를 적용한 경우, interleaving을 위한 BRO 연산의 수행 결과를 행렬의 형태로 표현하여 도시하는 도면.

도 5는 본 발명에 따라 순방향 패킷 데이터 트래픽에 대해 서브블럭 디인터리빙을 수행하기 위한 장치의 구성도.

도 6은 <예 1>에 제시된 부호화 패킷의 크기와 인터리빙 파라미터들을 적용하여 구성된 부호심볼들의 행렬을 나타낸 도면.

도 7은 상기 <예 1>의 경우 부호심볼 인덱스에 대해 생성된 행 인덱스와 임시주소와 주소 보상값 및 최종 읽기주소를 나타낸 도면.

도 8은 <예 2>에 제시된 부호화 패킷의 크기와 인터리빙 파라미터들을 적용하여 구성된 부호심볼들의 행렬을 나타낸 도면.

도 9는 상기 <예 2>의 경우 부호심볼 인덱스에 대해 생성된 행 인덱스와 임시주소와 주소 보상값 및 최종 읽기주소를 나타낸 도면.

도 10은 부호화 패킷의 크기가 408, 792, 1560, 3096인 경우, 본 발명의 일 실시예에 따른 읽기주소 발생기의 상세 구성을 도시한 도면.

도 11은 부호화 패킷의 크기가 2328인 경우, 본 발명의 일 실시예에 따른 읽기주소 발생기의 상세 구성을 도시한 도면.

도 12는 부호화 패킷의 크기가 3864인 경우, 본 발명의 일 실시예에 따른 읽기주소 발생기의 상세 구성을 도시한 도면.



## 【발명의 상세한 설명】

## 【발명의 목적】

## 【발명이 속하는 기술분야 및 그 분야의 종래기술】

- <15>        본 발명은 고속 패킷 데이터 통신 시스템에 관한 것으로서, 특히 인터리빙되어 전송된 데이터를 수신하여 원래의 형태로 디인터리빙하는 방법 및 장치에 관한 것이다.
- <16>        전형적인 디지털 이동통신 시스템, 특히 동기식 CDMA(Code Division Multiple Access) IS-2000 및 비동기식 UMTS(Universal Mobile Telecommunication Service) W(Wide)-CDMA와 같은 CDMA 방식의 이동통신 시스템은 음성 서비스와 회선 데이터(Circuit Data) 서비스를 통합적으로 지원하였다.
- <17>        동기식 CDMA(Code Division Multiple Access)2000 Release A/B 및 비동기식 UMTS(Universal Mobile Telecommunication Service)와 같은 이동통신 시스템에서는 멀티미디어 데이터의 신뢰성 있는 전송을 위해 터보 부호화와 인터리빙을 사용한다. 터보 부호화는 낮은 신호대 잡음비에서도 비트 에러율(Bit Error Rate: BER) 관점에서 매우 양호한 정보 복원 성능을 보이는 것으로 알려져 있다. 인터리빙은 페이딩 환경에서 비트들의 손상이 한곳에 집중되지 않고 여러 곳으로 분산되도록 한다. 이러한 인터리빙은 인접한 비트들이 랜덤 하게 페이딩 영향을 받도록 함으로서 군집에러(burst error)가 발생하지 않도록 해 주어 채널 부호화의 효과를 보다 높여준다.
- <18>        한편, 인터넷과 동영상 등 고속 패킷 데이터 전송을 필요로 하는 서비스에 대한 사용자의 요구가 증대됨에 따라 이동통신 시스템은 고속의 패킷데이터 서비스를 지원하는 형태로 발전하고 있는 추세이다. 3GPP 및 3GPP2에서 추진중인 CDMA2000 Release C, 이른 바



1xEV-DV(Evolution in Data and Voice) 표준안에 따르면, 기지국은 전송하고자 하는 패킷 데이터의 열을 채널 부호기에 의해 부호화하여 생성한 부호심볼들을 소정 크기의 서브블럭(sub-block)들로 분할하고, 상기 분리된 서브블럭들을 각각 인터리빙한다. 단말기는 상기 인터리빙된 데이터의 열을 수신하여 부호심볼의 형태로 변환한 후 기지국에서 사용한 인터리빙의 역과정인 디인터리빙을 거쳐, 채널 부호기의 출력 순서와 동일한 순서로 복원한다.

- <19> CDMA2000 Release A/B 표준에서는 인터리빙의 대상이 되는 부호심볼 열의 길이가 2를 밑으로 하고 비트 천이값(Bit Shift)를 지수로 하는 수의 배수가 되도록 규정하고 있는데 반해, 1xEV-DV에서는 이전 표준들과는 다른 새로운 인터리빙 규칙을 정의하고 있다. 따라서 1xEV-DV 표준의 경우, 단말기에서 상기 새로운 인터리빙 규칙에 따라 인터리빙된 데이터 열을 수신하여 신속하고도 효과적으로 디인터리빙하기 위한 기술을 필요로 하게 되었다.

#### 【발명이 이루고자 하는 기술적 과제】

- <20> 따라서 상기한 바와 같이 동작되는 종래 기술의 문제점을 해결하기 위하여 창안된 본 발명의 목적은, 통신시스템의 송신단에서 인터리빙되어 전송된 데이터를 수신단에서 고속으로 복구하는 방법 및 장치를 제공하는 것이다.
- <21> 본 발명의 다른 목적은 통신시스템의 수신단에서 순방향 패킷 데이터에 대해 서브블럭 디인터리빙을 수행하는 방법 및 장치를 제공하는 것이다.
- <22> 본 발명의 다른 목적은 통신시스템의 수신단에서 복호기의 입력버퍼를 이용하여 디인터리빙을 수행하는 방법 및 장치를 제공하는 것이다.

- <23> 본 발명의 다른 목적은 통신시스템에서 인터리빙된 데이터를 수신하여 저장하고, 디인터리빙을 위한 순서대로 독출하는 방법 및 장치를 제공하는 것이다.
- <24> 본 발명의 다른 목적은 1xEV-DV 통신시스템에서 인터리빙된 데이터를 수신하여 저장하고, 디인터리빙 규칙에 따라 생성된 독출 어드레스에 따라 읽어내는 방법 및 장치를 제공하는 것이다.
- <25> 상기 목적들을 달성하기 위한 본 발명의 방법은, 인터리빙된 순서대로 버퍼에 저장된 한 서브블럭의 부호심볼들을 디인터리빙된 순서대로 읽어내어 채널 복호기로 입력하는 방법에 있어서, 여기서 상기 부호심볼들은 상기 서브블럭의 크기에 따라 미리 정해지는 개수의 행과 열을 가지는 행렬의 형태로 저장되어 있고,
- <26> 상기 행렬의 마지막 열을 고려하지 않고, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 임시주소를 생성하는 과정과, 상기 행렬의 마지막 열을 고려하여 상기 임시주소를 보상하기 위한 주소 보상값을 계산하는 과정과, 상기 임시주소에 상기 주소 보상값을 더하여, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 읽기주소를 생성하는 과정을 포함한다.
- <27> 본 발명의 장치는, 인터리빙된 순서대로 버퍼에 저장된 한 서브블럭의 부호심볼들을 디인터리빙된 순서대로 읽어내어 채널 복호기로 입력하는 장치에 있어서, 여기서 상기 부호심볼들은 상기 서브블럭의 크기에 따라 미리 정해지는 개수의 행과 열을 가지는 행렬의 형태로 저장되어 있고,
- <28> 상기 행렬의 마지막 열을 고려하지 않고, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 임시주소를 생성하는 임시주소 생성기와, 상기 행렬의 마지막

열을 고려하여 상기 임시주소를 보상하기 위한 주소 보상값을 계산하는 주소 보상기와, 상기 임시주소에 상기 주소 보상값을 더하여, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 읽기주소를 생성하는 가산기를 포함한다.

### 【발명의 구성 및 작용】

- <29> 이하 첨부된 도면을 참조하여 본 발명의 바람직한 실시예에 대한 동작 원리를 상세히 설명한다. 하기에서 본 발명을 설명함에 있어 관련된 공지 기능 또는 구성에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략할 것이다. 그리고 후술되는 용어들은 본 발명에서의 기능을 고려하여 정의된 용어들로서 이는 사용자, 운용자의 의도 또는 관례 등에 따라 달라질 수 있다. 그러므로 그 정의는 본 명세서 전반에 걸친 내용을 토대로 내려져야 할 것이다.
- <30> 후술되는 본 발명은 인터리빙된 패킷 데이터의 열을 수신하여 디인터리빙하는 것이다. 특히 본 발명은 동기식 CDMA(Code Division Multiple Access) 통신방식의 하나인 CDMA2000 1xEV-DV(Evolution in data and voice) 표준안에 따라 인터리빙된 패킷 데이터의 열을 디인터리빙한다.
- <31> 먼저, CDMA2000 1xEV-DV 이동통신 시스템에서 패킷 데이터의 열을 인터리빙하고, 인터리빙된 데이터의 열을 디인터리빙하기 위한 구성 및 그 동작을 설명하기로 한다.
- <32> 도 1은 순방향 패킷 데이터 채널을 위한 패킷 데이터를 인터리빙하여 송신 서브패킷(Transmission Subpacket)을 생성하는 기지국 송신기의 구성을 나타낸 것이다.

- <33>        상기 도 1을 참고하면, 터보 부호기(Turbo Encoder)(110)는 입력되는 패킷 데이터의 열을 부호화 패킷(Encoder Packet) 단위로 소정 부호율(Code Rate) R에 따라 부호화하여 부호심볼들(Code Symbols)의 열을 출력한다. 여기서 1xEV-DV 표준에 의해 규정된 1/5의 부호율 R을 가지는 상기 터보 부호기(110)는 하나의 터보 인터리버(Turbo Interleaver)와 2개의 구성 부호기들(Constituent Encoders)로 구성된다.
- <34>        상기 터보 부호기(110)의 동작을 간략히 설명하면 하기와 같다. 입력되는 부호화 패킷은 변경없이 그대로 시스티메틱 서브블럭(Systematic Subblock) S로서 출력되며, 상기 제1 구성 부호기는 상기 입력 부호화 패킷을 부호화하여 2개의 패리티 서브블럭들(Parity Subblock) P0, P1을 출력한다. 상기 중간 인터리버는 상기 입력 부호화 패킷을 인터리빙하여 다른 시스티메틱 서브블럭 S'을 생성한다. 그러면 상기 제2 구성 부호기는 상기 시스티메틱 서브블럭 S'를 부호화하여 다른 2개의 패리티 서브블럭들 P'0, P'1을 출력한다. 상기 다른 시스티메틱 서브블럭 S'은 사용되지 않으므로, 터보 부호기(110)는 상기 입력 부호화 패킷과 동일한 크기를 가지는 상기 5개의 서브블럭들 S, P0, P1, P'0, P'1을 출력한다.
- <35>        채널 인터리버(Channel Interleaver)(120)는 서브패킷을 형성하는 부호심볼들이 높은 수준의 부호화이득(coding gain)을 보장할 수 있도록, 상기 터보 부호기(110)로부터 출력되는 부호심볼들을 적절히 배열한다. 이러한 배열은 심볼 분리(symbol separation), 서브블럭 인터리빙(subblock interleaving), 서브블럭 심볼 그룹화(subblock symbol grouping)를 차례로 수행함으로써 이루어진다. 보다 상세히 설명하면, 심볼 분리기(122)는 상기 터보 부호기(122)로부터 출력되는 다섯 종류의 부호심볼들을 종류대로 분류하여 다섯 개의 서브블럭들을 구성하고, 서브블럭 인터리버(124)는 상기 구성된 서브블럭들 각각을 동일한 인터리빙 규칙(interleaving rule)에 따라 인터리빙하며, 서브블럭 심볼 그룹화기(126)는 상기 인터리빙된 서브블럭들 중

일부의 부호심볼들을 서로 엇갈리도록(alternately) 그룹화한다. 서브패킷 심볼 선택기(Subpacket Symbol Selector)(130)는 상기 채널 인터리버(120)로부터의 부호심볼들 전체 또는 일부를 선택하여 서브패킷을 구성한다.

<36> 도 2에 채널 인터리버(120)에 의한 서브블럭 인터리빙과 서브블럭 심볼 그룹화 동작을 도시하였다. 도시된 바와 같이 종류별로 분리된 다섯 개의 서브블럭들 S, P0, P'0, P1, P'1(10 내지 18)은 서브블럭 인터리빙에 의해 각각 인터리빙된 서브블럭들(20 내지 28)이 된다. 서브블럭 심볼 그룹화에 의하면, 상기 인터리빙된 서브블럭들(20 내지 28) 중 인터리빙된 시스티메틱 서브블럭(20)은 변형되지 않으며, 인터리빙된 패리티 서브블럭들(22 내지 28)은 서로 엇갈리도록 그룹화된다. 즉 인터리빙된 서브블럭 P0(22)과 인터리빙된 서브블럭 P'0(24)의 부호심볼들을 서로 엇갈리도록 배치하여 제1 데이터 그룹(32)을 생성하고, 인터리빙된 서브블럭 P1(26)과 인터리빙된 서브블럭 P'1(28)의 부호심볼들을 서로 엇갈리도록 배치하여 제2 데이터 그룹(34)을 생성한다.

<37> 서브패킷 심볼 선택기(Subpacket Symbol Selector)(140)는 미리 정해진 선택 패턴에 따라, 상기 채널 인터리버(120)로부터의 인터리빙된 데이터 중 일부 심볼들을 선택하여 송신 서브패킷을 생성한다. 상기 생성된 서브패킷은 소정 변조방식을 가지고 변조된 후, 확산, 주파수 변환 등의 알려진 절차를 거쳐 단말기로 전송된다.

<38> 순방향 패킷 데이터 채널을 통해 기지국이 송신하는 고속의 패킷 데이터를 수신하는 단말은 상기 도 1에 나타난 동작을 역으로 수행함으로써 복호된 데이터를 얻는다.

<39> 도 3은 순방향 패킷 데이터 채널을 통해 수신된 서브패킷으로부터 복호된 데이터를 얻는 단말 수신기의 구성을 나타낸 것이다.

- <40>        상기 도 3을 참조하면, 서브패킷 제로 삽입기(Subpacket Zero Inserter)(210)는 기지국 송신기의 심볼 선택기(140)에 대응하여, 수신된 서브패킷의 정해진 위치에 '0'을 삽입한다. 기지국 송신기의 채널 인터리버(120)에 대응되는 단말기 수신기의 채널 디인터리버(channel deinterleaver)(220)는 서브블럭 심볼 역그룹화(subblock symbol ungrouping), 서브블럭 디인터리빙(Subblock deinterleaving), 심볼 역분리(symbol deseparation)를 차례로 수행한다.
- <41>        보다 상세히 설명하면, 서브블럭 심볼 역그룹화기(222)는 상기 서브패킷 제로 삽입기(210)로부터의 출력을 종류별로 분리하여 다섯 개의 인터리빙된 서브블럭들을 출력한다. 서브블럭 디인터리버(224)는 상기 인터리빙된 서브블럭들 각각을 기지국 송신기의 서브블럭 인터리버(124)에서 사용된 인터리빙 규칙에 대응하는 디인터리빙 규칙에 따라 디인터리빙하고, 심볼 역분리기(226)는 상기 디인터리빙된 출력을 결합하여 채널 복호기(Channel Decoder)(230)로 제공한다. 상기 채널 복호기(230)는 기지국 송신기의 터보 부호기(110)에서와 동일한 부호율 R을 가지고 상기 디인터리빙된 출력을 복호하여 복호된 데이터를 생성한다.
- <42>        상기와 같이 동작하는 단말 수신기에서, 서브블럭 디인터리빙은 서브블럭 인터리빙을 위해 사용된 인터리빙 규칙을 반영하여 수행되어야 하며, 이러한 인터리빙은 그 대상이 되는 입력 시퀀스(input sequence)의 길이에 따라 복잡도가 달라지게 된다.
- <43>        단말 수신기의 복잡도를 감소시키고 데이터 처리 속도를 향상시키기 위한 관점에서 볼 때, 서브블럭 디인터리빙의 수행은 두 가지 방법으로 달성될 수 있다. 첫 번째 방법은 채널 복호기의 입력버퍼에 부호심볼들을 저장할 때 디인터리빙을 수행하는 것으로서, 이 때 부호심볼들을 저장하기 위한 입력버퍼의 쓰기 주소들(write addresses)은 디인터리빙 규칙을 고려하여 생성된다. 두 번째 방법은 채널 복호기가 입력버퍼로부터 부호심볼들을 읽어올 때 디인터리빙



을 수행하는 것으로서, 이 때 부호심볼들을 읽어오기 위한 입력버퍼의 읽기주소들(read addresses)은 디인터리빙 규칙을 고려하여 생성된다.

<44>       상기 방법들 중 첫 번째는 쓰기 주소의 생성시에 서브블럭 디인터리빙 뿐 아니라 복조과정에 수반되는 디서플링을 함께 반영해야 하므로 변조방식의 종류 및 디서플링 패턴에 따라 부가적인 제어 장치가 요구된다. 따라서, 본 명세서에서는 서브블럭 디인터리빙을 수행함에 있어서 채널 복호기의 입력버퍼에서 부호심볼들을 읽어 오기 위한 읽기주소들을 생성하는 방법에 대해 기술할 것이다.

<45>       1xEV-DV 표준에 따르면, 순방향 트래픽을 전달하는 순방향 패킷 데이터 채널(Forward packet data channel: F-PDCH)에서 사용 가능한 부호화 패킷(Encoder Packet: EP)의 크기는 6가지로 규정되어 있다. 채널 부호기로부터 출력되는 부호심볼들은 서브블럭 별로 인터리빙을 거치게 되는데, 이 때 각 서브블럭의 크기는 부호화 패킷의 크기와 동일하므로 결국 서브블럭 인터리빙/디인터리빙의 대상이 되는 부호심볼열의 길이 또한 부호화 패킷의 크기와 같다.

<46>       하기의 <표 1>은 1xEV-DV 표준에 규정된 부호화 패킷의 크기들과 그에 따른 서브블럭 인터리빙/디인터리빙 파라미터들을 보여 준다. 상기 <표 1>에서 비트천이값(bit shift)  $m$ 과 상한값(up limit)  $J$ 는  $N_{EP}$ 의 크기를 가지는 부호심볼들의 열을 행렬의 형태로 배열할 때, 행(row)의 개수 및 열(column)의 개수와 각각 관련되며, 나머지값(remainder)  $R$ 은 상기 행렬의 마지막 열이 가지는 원소의 개수를 의미한다.

<47>

【표 1】

EP Size ( $N_{EP}$ )	Bit Shift (m)	Up Limit (J)	Remainder (R)
408	7	4	24
792	8	4	24
1560	9	4	24
2328	10	3	280
3096	10	4	24
3864	11	2	1816

<48> 결과적으로, 부호화 패킷의 크기  $N_{EP}$ 와 인터리빙/디인터리빙 파라미터들은 하기의 <수학식 1>과 같은 관계에 있다.

<49> 【수학식 1】  $N_{EP} = 2^m \times (J-1) + R$

<50> 본 발명에 따른 디인터리빙 규칙을 설명하기 위해, 먼저 인터리빙된 데이터의 열의 형태를 일 예를 들어 상세히 설명한다. 도 4는  $N_{EP}=408$ 인 경우 하나의 서브블럭에 대해 수행되는 인터리빙 과정을 세 단계로 분리하여 도시한 것이다.

<51> 도시된 바와 같이, 단계(40)에서 408개의 부호심볼들은 채널 부호기의 출력 부호심볼 인덱스  $k$  ( $k=0, 1, 2, \dots, 407$ )에 따라 순차적으로 버퍼에 저장(writing)된다.(40) 여기서 화살표들은 부호심볼들이 저장되는 순서(하->우)를 나타내는 것으로서 도시된 바와 같이 열->행의 순서로 저장된다. 단계(42)에서 상기 저장된 부호심볼들은 각 행별로 행 번호의 비트 역전환(Bit Reversal Order)에 의해 재배열된다. 여기서 비트 역전환이란 비트들의 순서를 역전시키는 것을 말한다.

<52> 일 예로서, 행 번호가 "0000001(1)"인 행은 비트 역전환에 의하여 행 번호가 "1000000(64)"인 행으로 이동한다. 단계(44)에서 상기 행 단위 비트 역전환(Row BRO)에 의해 상

기 재배열된 부호심볼들은 행->열의 순서로 버퍼로부터 출력된다. 여기서 화살표들은 부호심볼들이 출력되는 순서(우->하)를 나타낸 것이다.

<53>      상기 도 4를 참조하면, 버퍼에 저장되는 부호심볼들의 행렬은  $2^7$ 개의 행들과 4개의 열들로 구성되고 마지막 네 번째 열은 단지 24개의 원소만을 가진다. 여기서 만약 상기 네 번째 열이 원소를 하나도 가지지 않는다면, 즉  $R=0$ 이라면, 기지국 송신기에서 서브블럭 인터리빙은  $i$  ( $i=0, 1, 2, \dots, 407$ ) 번째 읽어내는 부호심볼의 읽기주소  $B_i$ 를 하기의 <수학식 2>와 같이 생성하여 해당 주소의 부호심볼을 버퍼로부터 읽어내는 것만으로 달성될 수 있다.

$$\begin{aligned} B_i &= 2^m \times (i \bmod (J-1)) + BRO_m([i/(J-1)]) \\ &= 2^7 \times (i \bmod 3) + BRO_7([i/3]) \end{aligned}$$

【수학식 2】

<55>      여기서 " $i$ "는 서브블럭 인터리버의 출력 부호심볼 인덱스이고, " $\bmod$ "는 모듈로(Modulus) 연산을 의미하고, " $BRO_7(\cdot)$ "은 7비트 단위 비트 역전환(Bit Reverse Ordering) 연산을 의미하며, " $[\cdot]$ "는 입력( $\cdot$ )을 초과하지 않는 최대의 정수를 의미한다.

<56>      한편, 기지국에서 인터리빙된 부호심볼들은 변조, 확산 등을 알려진 절차를 거쳐 단말로 수신된다. 단말이 수신한 부호심볼들은 채널 복호기의 입력버퍼에 서브블럭 인터리빙된 순서대로 저장된다. 따라서 서브블럭 디인터리빙은 채널 복호기의 입력 부호심볼 인덱스  $k$  ( $k=0, 1, 2, \dots, 407$ )에 대해  $k$ 번째 읽어 내는 부호심볼의 읽기주소  $A_k$ 를 상기 <수학식 2>의 역함수에 따라 생성하여 해당 주소의 부호심볼을 채널 복호기 입력버퍼로부터 읽어 냄으로써 달성될 수 있다. 하기의 <수학식 3>에 상기 <수학식 2>의  $B_i$ 의 역함수를 이용한 읽기주소  $A_k$ 의 생성식을 나타내었다.

$$\begin{aligned} A_k &= (J-1) \times BRO_m(k \bmod 2^m) + [k/2^m] \\ &= 3 \times BRO_7(k \bmod 2^7) + [k/2^7] \end{aligned}$$

【수학식 3】

- <58>      상기 <수학식 2> 및 상기 <수학식 3>은 앞서 언급한 바와 같이  $R=0$ 인 경우에 대해 고려한 것이다. 그러나,  $N_{EP}=408$ 인 경우를 포함하여 상기 <표 1>에서 볼 수 있듯이 규정된 모든 부호화 패킷 크기에 대해  $R$ 은 0이 아닌 양의 값을 가지므로, 서브블럭 디인터리빙을 위한 읽기주소 생성식은 상기 <수학식 3>과 같이 간단하지 않다.
- <59>      따라서, 본 발명에서는 채널 복호기의 입력버퍼를 위한 읽기주소(read address)의 생성을, 임시주소 생성(interim address generation)과 주소 보상(address compensation)과 가산(adding process)의 세 단계로 분리하여 처리한다. 여기서 임시주소(Interim Address)란  $R=0$ 으로 가정하여 생성한 읽기주소를 의미하며, 상기 주소 보상이란 상기 임시주소를 보상하여 최종 읽기주소를 생성하기 위한 주소 보상값(Address Compensation Factor)을 생성하는 동작이다.
- <60>      도 5는 본 발명에 따라 순방향 패킷 데이터 트래픽에 대해 서브블럭 디인터리빙을 수행하기 위한 장치를 도식화한 것이다. 여기서 도시한 장치는 도 3에 나타낸 단말 수신기에서 서브블럭 심볼 역그룹화기(222)로부터 출력되는 인터리빙된 서브블럭들 각각에 대하여 동작하는 것이다.
- <61>      상기 도 5를 참조하면, 한 서브블럭의 인터리빙된 부호심볼들은 채널 복호기(330)를 위한 입력버퍼(Input Buffer)(310)에 인터리빙된 순서대로 저장된다. 여기서 상기 입력버퍼(310)의 메모리 구조는 부호화 패킷의 크기  $N_{EP}$ 에 따라  $2^m \times J$ 의 메모리 배열로 형성되며, 부호심볼들은 상기 메모리 배열내에 행->열의 순서로 저장된다. 그러면 디인터리빙 주소 발생기(320)는 기지국 송신기에서 사용한 인터리빙 규칙에 대응하는 디인터리빙 규칙에 따라 상기 입력버퍼(310)에서 부호심볼들을 읽어내기 위한 읽기주소들을 생성한다.
- <62>      하기의 <수학식 4>에 상기 디인터리빙 주소 발생기(320)에서 읽기주소를 생성하기 위한 생성식을 나타내었다.

<63> 【수학식 4】  $A_k = IA_k + C(r_k)$

<64> 여기서 " $k$ "는 채널 복호기(330)에서 입력을 요청한 부호심볼의 인덱스이고, " $A_k$ "는 서브블럭 디인터리빙을 반영한 입력버퍼(310)의 읽기주소이고, " $IA_k$ "는 임시주소이고, " $C(r_k)$ "는 상기 임시주소를 위한 주소 보상값이고, " $r_k$ "는 인터리빙된 부호심볼 행렬의 행 인덱스(Row Index)이다.

<65> 보다 상세히 설명하면, 임시주소 생성기(Interim Address Generator)(322)는 채널 복호기로부터 부호심볼 인덱스  $k$ 를 입력받아, 상기  $k$ 에 대하여 마지막 열을 고려하지 않은 임시주소  $IA_k$ 와 행 인덱스  $r_k$ 를 생성한다. 주소 보상기(334)는 상기 행 인덱스  $r_k$ 를 입력으로 받아 해당 행에 대한 주소 보상값  $C(r_k)$ 를 생성하며, 가산기(326)는 상기 임시주소  $IA_k$ 에 해당 행의 주소 보상값  $C(r_k)$ 를 더하여 최종 읽기주소  $A_k$ 를 상기 입력버퍼(310)로 제공한다.

<66> 그러면 상기 입력버퍼(310)에서 상기 읽기주소  $A_k$ 에 해당하는 부호심볼이 채널 복호기(330)로 출력된다. 이러한 일련의 절차는 상기 입력버퍼(310)에 저장된 모든 부호심볼들이 읽어내어질 때까지 계속된다.

<67> 상기 임시주소 생성기(322)에 의한 임시주소의 생성에 대해 보다 상세히 설명하면, 도 5에서 임시주소 생성기(322)는 채널 복호기(330)로부터 입력받고자 하는 부호심볼의 인덱스  $k$ 를 입력받아 임시주소  $IA_k$ 와 해당 행 인덱스  $r_k$ 를 출력한다. 한 서브블럭의 인터리빙된 부호심볼들을 저장하는 입력버퍼(310)의 메모리 배열을 도 4의 단계(44)에 나타낸 행렬과 같은 형태라고 하면,  $k$ 번째 부호심볼의 행 인덱스  $r_k$ 는 하기의 <수학식 5>와 같이 간단히 계산된다.

<68> 【수학식 5】  $r_k = BRO_m(k \bmod 2^m)$

<69> 여기서 " $m$ "은 앞서 언급한 <표 1>에 나타낸 비트 천이값(bit shift)을 의미한다.

<70> 앞서 언급한 바와 같이, 임시주소는 상기 <표 1>에서 나머지값  $R$ 을 0 또는 행의 총 개수( $=2^m$ )와 같다고 가정하여 생성한 임시 읽기주소이다. 이 경우 입력버퍼(310)에 저장되는 인터리빙된 부호심볼들은  $2^m \times (J-1)$  또는  $2^m$ 의 크기를 가지는 완전한 형태의 행렬인 것으로 한다.

<71> 상기 <표 1>에 따르면  $N_{EP}$ 가 3096 이하인 다섯 가지 종류의 부호화 패킷들에 대해서는  $R$ 이 행의 총 개수의 절반인  $2^{m-1}$ 보다 작은 반면,  $N_{EP}$ 가 3864인 부호화 패킷에 대해서는  $R$ 이 행의 총 개수의 절반인  $2^{m-1}$ 보다 크다는 사실을 확인할 수 있다.  $R$ 이 행 개수의 절반보다 작은 경우  $2^m \times (J-1)$ 의 크기를 가지는 행렬을 사용하면 주소 보상시에는  $R$ 개의 원소들만을 더 고려하면 된다. 반면  $R$ 이 행 개수의 절반보다 큰 경우  $2^m$ 의 크기를 가지는 행렬을 사용하면 주소 보상시에는  $(2^m - R)$ 개의 원소들만을 제외하면 된다. 따라서 주소 보상기(324)의 연산을 보다 간단히 하기 위한 임시주소 생성식은 하기의 <수학식 6>과 같이 나머지값  $R$ 의 크기에 따라 두 가지로 구분된다.

<72>

$$\begin{aligned} &\text{if } R < 2^{m-1} \\ &\quad IA_k = (J-1) \cdot BRO_m(k \bmod 2^m) + \lceil k/2^m \rceil \\ &\quad \quad = (J-1) \cdot r_k + \lceil k/2^m \rceil \\ &\text{otherwise} \\ &\quad IA_k = J \cdot BRO_m(k \bmod 2^m) + \lceil k/2^m \rceil \\ &\quad \quad = J \cdot r_k + \lceil k/2^m \rceil \end{aligned}$$

【수학식 6】

<73> 다음으로 상기 주소 보상기(324)에 의한 주소 보상에 대해 보다 상세히 설명하면, 주소 보상기(324)는 임시주소 생성기(322)로부터 행 인덱스  $r_k$ 를 입력받아 주소 보상값  $\alpha(r_k)$ 를 출력한다. 앞서 언급한 바와 같이, 부호화 패킷의 크기에 따른 나머지값  $R$ 이 행 개수의 절반  $2^{m-1}$ 보다 작은 경우 더 고려해야할 원소가 발생할 때마다 주소 보상값은 1씩 증가한다. 반면

나머지값  $R$ 이 행 개수의 절반보다 큰 경우 제외해야할 원소가 발생할 때마다 주소 보상값은 1씩 감소한다.

- <74>      상기 주소 보상값 생성의 원리를, 하기에 기술될 두 가지 실시예를 통해 설명하도록 한다.
- <75>      <예 1 :  $N_{EP}=20$ ,  $m=4$ ,  $J=2$ ,  $R=4$ >
- <76>      상기 <예 1>는 나머지값  $R$ 이 행 개수의 절반인  $8(=2^m-1)$ 보다 작은 경우이다.
- <77>      도 6은 상기 <예 1>에 제시된 부호화 패킷의 크기와 인터리빙 파라미터들을 적용하여 구성된 부호심볼들의 행렬을 나타낸 것으로서 도시된 바와 같이, 첫 번째 열은 16개의 원소들을 모두 가지지만 두 번째 열은 4개의 원소들만을 가진다.
- <78>      row BRO 연산 이전의 행렬에서 마지막 열의 부호심볼들은 각각 행 인덱스 0,1,2,3에 위치하고 있지만, row BRO 연산 이후 마지막 열의 부호심볼들은 서로 동일한 거리  $4(=16/4)$ 를 유지하면서 고르게 분산되어, 각각 행 인덱스 0, 4, 8, 12로 이동한다. 이러한 분산은 BRO 연산의 특성 때문에 나타나는 것으로서 이 특성을 이용하면  $R$ 이  $2^m-1$  보다 작은 경우의 주소 보상값은 마지막 열에서 최초로 부호심볼이 나타난 이후 매 4번째 행마다 1씩 증가됨을 알 수 있다.
- <79>      도 7은 상기 <예 1>의 경우 부호심볼 인덱스(C.S. Index)  $k$ 에 대해 생성된 행 인덱스 (Row Index)

$r_k$ 와 임시주소  $IA_k$ 와 주소 보상값  $C(r_k)$  및 최종 읽기주소  $A_k$ 를 차례로 나타낸 것이다. 도시된 바와 같이 최종 읽기주소  $A_k$ 는 임시주소  $IA_k$ 에 주소 보상값  $C(r_k)$ 를 더한 것이며, 주소 보상값  $C(r_k)$ 는 마지막 열에서 부호심볼이 나타난 다음 행마다 1씩 증가한다. 이것은 임시주소 생성시 고려되지 않았던 마지막 열에 추가적으로 고려해야할 부호심볼이 삽입되면, 상기 삽입된 부호심볼을 위한 읽기주소가 추가로 할당되고 이에 따라 이후의 임시주소들이 하나씩 밀려야 하기 때문이다.

<80> 결국 주소 보상값  $C(r_k)$ 는 채널 부호기(330)가 요청한 인덱스  $k$ 를 가지는 부호심볼이 입력버퍼(310)에 저장된 행렬의 어떤 행에 속해 있는지 및 해당 부호심볼이 속한 행 이전에 임시주소 생성시 고려되지 않았던 부호심볼들이 삽입되는 행들이 몇개인지를 파악함으로써 계산될 수 있다.

<81> 예를 들어, 부호심볼 인덱스  $k$ 가 10인 부호심볼의 행 인덱스는 5이고, 이 행 이전에는 임시주소 생성시 고려되지 않았던 부호심볼들이 삽입된 행들의 개수가 2개(행 인덱스=0,4)이므로, 부호심볼 인덱스 10에 대한 주소 보상값은 2가 되는 것이다. 상기 설명한 주소 보상값의 계산 과정을 정리하면 하기의 <수학식 7>과 같다.

<82> **【수학식 7】** 
$$C_d^+(r_k) = \left[ \frac{r_k + d - (r^+ + 1)}{d} \right]$$

<83> 여기서 "d"는 행의 총 개수를 삽입될 부호심볼의 개수 혹은 부호심볼이 삽입될 행의 개수로 나눈 값으로서 마지막 열에 속한 부호심볼들의 행간 거리를 의미하며, " $r^+$ "는 최초로 삽입되는 부호심볼이 위치한 행의 인덱스를 의미한다. 마지막 열에 최초로 삽입되는 부호심볼은 BRO 연산의 특성상 항상 행 인덱스 0에 위치하게 되므로 이 경우  $r^+$ 는 0이다. 하지만, 나머지 값 R 및 행의 개수에 따라 마지막 열에 이미 삽입된 부호심볼이 존재하는 경우, 상기 이미 삽

입되어 있던 부호심볼의 행 인덱스에 따라 추가로 삽입되는 첫 번째 부호심볼의 행 인덱스  $r^+$ 가 결정된다. 상기 <수학식 7>의 주소 보상값  $C_d^+$ 에서 "+"는 마지막 열에 부호심볼이 "삽입"되는 경우임을 나타내기 위한 것이며, "d"는 부호심볼이 삽입되는 행에 대해 전체 행 개수의 비율을 나타내기 위한 것이다.

<84> 상기 <예 1>에 대해 상기 <수학식 7>을 적용해 보면 상기 <예 1>을 위한 주소 보상값 생성식은 하기의 <수학식 8>과 같다.

<85> **[수학식 8]** 
$$C_d^+(r_k) = \left\lceil \frac{r_k + 4 - (0 + 1)}{4} \right\rceil = \left\lceil \frac{r_k + 3}{4} \right\rceil$$

<86> 상기와 같이 예를 들어 부호심볼 인덱스  $k$ 가 10인 부호심볼의 행 인덱스는 5이므로 부호심볼 인덱스 10에 대한 주소 보상값은 2가 되는 것이다. 또한 상기 부호심볼의 임시주소는 5이므로 보상된 주소값은  $A_k$ 는 7이 된 것이다.

<87> <예 2 :  $N_{EP}=20, m=4, J=2, R=12$ >

<88> 상기 <예 2>는 나머지값  $R$ 이 행 개수의 절반인  $8(=2^m-1)$ 보다 크거나 같은 경우이다.

<89> 도 8은 상기 <예 2>에 제시된 부호화 패킷의 크기와 인터리빙 파라미터들을 적용하여 구성된 부호심볼들의 행렬을 나타낸 것으로서, 도시한 바와 같이 첫 번째 열은 16개의 원소들을 모두 가지지만 두 번째 열에서는 4개의 원소들이 제거되어야 한다.

<90> row BRO 연산 이전의 행렬에서, 마지막 열의 제거 대상 부호심볼들은 각각 행 인덱스 12,13,14,15에 위치하고 있지만, row BRO 연산 이후 제거 대상 부호심볼들은 서로 동일한 행간 거리  $4(=16/4)$ 를 유지하면서 분산되어, 각각 행 인덱스 3,7,11,15로 이동한다. 이러한 분산은 BRO 연산의 특성 때문에 나타나는 것으로서 이 특성을 이용하여

$R$ 이  $2^{m-1}$  보다 크거나 같은 경우의 주소 보상값은 마지막 열에서 최초로 부호심볼이 제거된 이후 매 4번째 행마다 1씩 감소됨을 알 수 있다.

<91> 도 9는 상기 <예 2>의 경우 부호심볼 인덱스(C.S. Index)  $k$ 에 대해 생성된 행 인덱스 (Row Index)  $r_k$ 와 임시주소  $IA_k$ 와 주소 보상값  $C(r_k)$  및 최종 읽기주소  $A_k$ 를 차례로 나타낸 것이다. 도시한 바와 같이, 최종 읽기주소  $A_k$ 는 임시주소  $IA_k$ 에 음의 값인 주소 보상값  $C(r_k)$ 를 더한 것이며, 주소 보상값  $C(r_k)$ 는 마지막 열에서 부호심볼이 제거된 다음 행마다 1씩 감소한다. 이것은 임시주소 생성시 함께 고려되었던 마지막 열로부터 고려하지 않았어야 할 부호심볼이 제거되면, 상기 제거된 부호심볼을 위해 할당되었던 읽기주소가 회수(deallocate)되고 이에 따라 이후의 임시주소들이 하나씩 당겨져야 하기 때문이다.

<92> 결국 주소 보상값  $C(r_k)$ 는 채널 부호기(330)가 요청하는 인덱스  $k$ 를 가지는 부호심볼이 입력버퍼(310)에 저장된 행렬의 어떤 행에 속해 있는지 및 해당 부호심볼이 속한 행 이전에 임시주소 생성시 함께 고려되었던 부호심볼들이 제거되는 행들이 몇 개인지를 파악함으로써 계산될 수 있다.

<93> 예를 들어, 부호심볼 인덱스  $k$ 가 13인 부호심볼의 행 인덱스는 11이고, 이 행 이전에 임시주소 생성시 함께 고려되었던 부호심볼들이 제거된 행들의 개수는 2개(행 인덱스=3,7)이므로, 부호심볼 인덱스 13에 대한 주소 보상값은 -2가 되는 것이다. 상기 설명한 주소 보상값의 계산 과정을 정리하면 하기의 <수학식 9>와 같다.

<94> **【수학식 9】** 
$$C_d(r_k) = - \left[ \frac{r_k + d - (r' + 1)}{d} \right]$$

<95> 여기서 "d"는 행의 총 개수를 제거될 부호심볼들의 개수 혹은 부호심볼들이 제거될 행들의 개수로 나눈 값으로서 마지막 열에서 제거될 부호심볼들의 행간 거



리를 의미하며, " $r^-$ "는 최초로 제거되는 부호심볼이 위치한 행의 인덱스를 의미한다. 마지막 열에서 최초로 제거되는 부호심볼은 BRO 연산의 특성상 항상 행 인덱스  $d-1$ 에 위치하게 되므로 이 경우  $r^-$ 는  $d-1$ 이다. 하지만, 나머지값 및 행의 개수에 따라 마지막 열이 완전히 채워지지 않은 상태에서는 이미 제거된 부호심볼이 존재하는 경우, 상기 이미 제거된 부호심볼의 행 인덱스에 따라 추가로 제거되는 첫 번째 부호심볼의 행 인덱스  $r^-$ 가 결정된다. 상기 <수학식 9>의 주소 보상값  $C_d^-$ 에서 " $-$ "는 마지막 열에서 부호심볼이 "제거"되는 경우임을 나타내며, " $d$ "는 부호심볼이 제거되는 행에 대해 전체 행 개수의 비율을 나타낸다.

<96> 상기 <예 2>에 대해 상기 <수학식 9>를 적용해 보면 상기 <예 2>를 위한 주소 보상값 생성식은 하기의 <수학식 10>과 같다.

<97> **【수학식 10】** 
$$C_4^-(r_k) = -\left[\frac{r_k + 4 - (3 + 1)}{4}\right] = -\left[\frac{r_k}{4}\right]$$

<98> 상기와 같이 예를 들어 부호심볼 인덱스  $k$ 가 13인 부호심볼의 행 인덱스는 11이므로 부호심볼 인덱스 10에 대한 주소 보상값은 -2가 되는 것이다. 또한 상기 부호심볼의 임시주소는 22이므로 보상된 주소값은  $A_k$ 는 20이 된 것이다.

<99> 상기 <예 1>과 상기 <예 2>를 통해 부호심볼이 삽입되는 경우와 가상 부호심볼이 제거되는 경우 각각에 대한 주소 보상값의 계산 과정을 설명하였다. 여기서 전체 행의 개수에 대해 삽입 혹은 제거되는 행들의 비율  $d$ 는 삽입될 부호심볼 혹은 제거될 부호심볼이 속한 행들 간의 거리를 나타내며, 하기의 <수학식 11>과 같이 정의된다.

<100> **【수학식 11】** 
$$d = \frac{\text{number of total rows}}{\text{number of rows to be considered}}$$

<101> 여기서  $d$ 는 자연수가 되어야 한다.

- <102> 여기서 만일 상기 " $d$ "가 자연수가 아닐 경우 상기 <예 1>과 상기 <예 2>에서 설명한 BRO 연산의 특성을 이용한 주소 보상값 생성식은 맞지 않게 된다. 상기 <수학식 11>에서 분자인 전체 행 개수는 항상 2의 거듭제곱 형태이므로,  $d$ 가 자연수가 되기 위해서는 분모인 삽입 혹은 제거 행의 개수 역시 2의 거듭제곱 형태가 되어야 한다. 앞서 언급한 <표 1>에서 나머지값  $R$ 은 부호심볼이 삽입되거나 제거되어야 하는 행들의 개수를 나타내는데, 모든 부호화 패킷의 크기에 대해 어떠한  $R$ 도 2의 거듭제곱 형태는 아니다. 그러나, 이러한 경우에도  $R$ 을 2의 거듭제곱들의 합으로 표현하면, 상기 <수학식 7> 및 상기 <수학식 9>를 적용할 수 있다.
- <103> 이하 규정된 부호화 패킷의 크기들  $N_{EP}=408, 792, 1560, 2328, 3096, 3864$  각각에 대하여 실제로 적용될 주소 보상값 생성 원리를 설명한다.
- <104> A.  $N_{EP}=408, 792, 1560, \text{ or } 3096$ 인 경우( $R=24$ ) 주소 보상값의 생성
- <105> 먼저 부호화 패킷의 크기가 408인 경우에 대해 설명하기로 한다. 여기서, 인터리빙된 부호심볼들은 도 4의 44와 같은 행렬로부터 행→열의 순서대로 읽혀져 복호기 입력버퍼에 저장되어 있다.
- <106> 이 때, 하나의 서브블럭에 포함된 408개의 부호심볼들은  $128(=2^7)$ 개의 행들을 가지는 행렬로 구성되고, 이 행렬의 마지막 열은 임시주소 생성시 고려되지 않았던 24개의 부호심볼들을 가지고 있다. 24를 2의 거듭제곱 형태로 다시 표현하면  $2^4+2^3$ 이 되므로, 상기 24개의 부호심볼들을 상위 16개와 나머지 8개로 나누어 고려하기로 한다.
- <107> 즉, 상위 16개의 부호심볼들은 최상위 행부터 소정 행간 거리  $8(=128/16)$  만큼씩 떨어진 행들 0, 8, 16, ..., 104, 112, 120에 배열되고, 다음으로 나머지 8개의 부호심볼들이  $4(=8/2)$



번째 행으로부터 소정 행간 거리 16(=128/8) 만큼씩 떨어진 행들, 4, 20, 36, ..., 84, 100, 116에 배열된다.

<108> 결국, 24개의 부호심볼들 중 16개의 부호심볼들은 행 인덱스가 8의 배수가 될 때마다 주기적으로 삽입되고 나머지 8개의 부호심볼들은 행 인덱스가 16의 배수가 될 때마다 주기적으로 삽입되므로, 24개의 부호심볼들은 16을 주기로 그 삽입 패턴이 동일하다고 할 수 있다. 따라서, 128개의 행들에 24개의 부호심볼들을 삽입하는 것은, 매 16(=128/8)개의 행들마다 3(=24/8)개의 부호심볼들을 삽입하는 것과 동일하다고 추약 해석할 수 있다.

<109> 상기 삽입해야할 부호심볼들의 개수 3을 2의 거듭제곱 형태로 표현하면,  $3=2+1$  혹은  $3=4-1$ 이다. 이 두 가지 경우에 대하여 주소 보상값을 각각 계산해 보면 다음과 같다.

<110> 먼저, 3을  $2+1$ 로 표현한 경우, 16개의 행들에 먼저 2개의 부호심볼들을 삽입하고 다시 1개의 부호심볼을 더 삽입해야 한다. 그러면 첫 번째 삽입시에  $d$ 는 8(=16/2)이고 두 번째 삽입시에  $d$ 는 16(=16/1)이며 이에 따라 주소 보상값  $C(r_k)$ 는 하기의 <수학식 12>와 같이 계산된다.

<111>

$$\begin{aligned}
 C(r_k) &= C_8^+(r_k) + C_{16}^+(r_k) \\
 &= \left[ \frac{r_k + 8 - (0+1)}{8} \right] + \left[ \frac{r_k + 16 - (4+1)}{16} \right] \\
 &= \left[ \frac{r_k + 7}{8} \right] + \left[ \frac{r_k + 11}{16} \right]
 \end{aligned}$$

【수학식 12】

<112> 상기의 <수학식 12>에서  $d$ 가 8인 경우는 행렬의 마지막 열이 부호심볼을 전혀 가지고 있지 않은 상태에서 행간 거리가 8인 2개의 부호심볼들을 상기 마지막 열에 최초로 삽입하는 것이므로, 첫 번째 삽입되는 부호심볼의 행 인덱스는 0고 따라서 이 때의  $r^+$ 는 0이다. 또한  $d$ 가 16인 경우는 행렬의 마지막 열이 이미 행간 거리가 8인 2개의 부호심볼들을 가지고 있는 상태에서 1개의 부호심볼을 추가로 삽입하는 경우인데, 여기서 추가되는 부호심볼은 BRO 연산의 특

성상 이미 삽입된 2개의 부호심볼들 사이의 중간에 삽입되므로, 따라서 이 때의  $r^+$ 는  $4(=(0+8)/2)$ 가 된다.

<113> 다음으로 3을 4-1로 표현한 경우, 16개의 행들에 먼저 4개의 부호심볼들을 삽입하고 삽입된 부호심볼들 중 1개의 부호심볼을 제거한다. 그러면 삽입시에  $d$ 는  $4(=16/4)$ 이고 제거시에  $d$ 는  $16(=16/1)$ 이며 이에 따라 주소 보상값  $C(r_k)$ 는 하기의 <수학식 13>과 같이 계산된다.

<114>

$$C(r_k) = C_4^+(r_k) + C_{16}^-(r_k)$$

$$= \left[ \frac{r_k + 4 - (0 + 1)}{4} \right] - \left[ \frac{r_k + 16 - (12 + 1)}{16} \right]$$

【수학식 13】  $= \left[ \frac{r_k + 3}{4} \right] - \left[ \frac{r_k + 3}{8} \right]$

<115> 하기의 <수학식 13>에서  $d$ 가 4인 경우 행렬의 마지막 열에 첫 번째 삽입되는 부호심볼의 행 인덱스  $r^+$ 는 0이며,  $d$ 가 16인 경우 마지막 열에서 최초로 제거되는 부호심볼의 행 인덱스  $r^-$ 는 12이다. 이는 BRO 연산의 특성상 마지막 열에 이미 삽입되어 있던 4개의 부호심볼들 중 가장 마지막 행의 부호심볼이 가장 먼저 제거되어야 하기 때문이다. 즉, 마지막 열에 이미 삽입되어 있던 부호심볼들의 개수는 4이고 그 행간 거리는 4이므로, 마지막 행의 인덱스  $r^-$ 는  $12(=0+3 \times 4)$ 이다.

<116> 상기 <수학식 12>와 상기 <수학식 13>의 연산 결과는 완전히 동일하다. 주소 보상기 (324)를 하드웨어로 구현함에 있어서,  $r_k$ 에 서로 다른 값 7과 11을 더하는 두 개의 가산기를 필요로 하는 상기 <수학식 12> 보다는,  $r_k$ 에 3을 더하는 한 개의 가산기만을 필요로 하는 상기 <수학식 13>이 구현 측면에서 보다 효율적이다.

<117> 부호화 패킷의 크기가 792, 1560, 3096인 경우에는 부호화 패킷의 크기가 408인 경우에 비해 행렬을 이루는 행의 개수가 각각 2배, 4배, 8배로 증가했을 뿐, 임시주소 생성시 고려되지 않았던 부호심볼의 개수는 24개로 모두 동일하다. 따라서, 이들 경우에는 상기 <수학식

13>을 이용하여 비트 천이값(bit shift)  $m$ 을 변수로 가지도록 일반화한 생성식을 사용하여 주소 보상값을 계산할 수 있다. 하기의 <수학식 14>는  $N_{EP}=408, 792, 1560, 3096$ 인 경우에 대해 적용되는 주소 보상값 생성식을 나타낸 것이며, 앞서 언급한 <표 1>에 나타낸 바와 같이 이때의  $m$ 은 각각 7, 8, 9, 10 이다.

<118>

$$\begin{aligned}
 C(r_k) &= C_{2^{m-5}}^+(r_k) + C_{2^{m-3}}^-(r_k) \\
 &= \left[ \frac{r_k + 2^{m-5} - (0+1)}{2^{m-5}} \right] - \left[ \frac{r_k + 2^{m-3} - (3 \cdot 2^{m-3} + 1)}{2^{m-3}} \right] \\
 &= \left[ \frac{r_k + 2^{m-5} - 1}{2^{m-5}} \right] - \left[ \frac{r_k + 2^{m-5} - 1}{2^{m-3}} \right]
 \end{aligned}$$

【수학식 14】

<119> 여기서 부호심볼 인덱스  $k$ 는 0, 1, ...,  $N_{EP}-1$  이고, 행 인덱스  $r_k$ 는 0, 1, ...,  $2^m-1$  이다

<120> B.  $N_{EP}=2328$ 인 경우( $R=280$ ) 주소 보상값의 생성

<121> 부호화 패킷의 크기가 2328인 경우, 하나의 서브블럭에 포함된 2328개의 부호심볼들은 1024( $=2^{10}$ ) 개의 행들을 가지는 행렬로 구성되고, 이 행렬의 마지막 열은 임시주소 생성시 고려되지 않았던 280개의 부호심볼들을 가지고 있다. 280을 2의 거듭제곱 형태로 다시 표현하면  $2^8+2^4+2^9$ 가 되므로, 상기 280개의 부호심볼들을 상위 256개와 다음 16개와 나머지 8개로 나누어 고려하기로 한다.

<122> 즉, 상위 256개의 부호심볼들은 행 인덱스가 4( $=1024/256$ )의 배수가 될 때마다 주기적으로 삽입되고, 다음 16개의 부호심볼들은 행 인덱스가 64( $=1024/16$ )의 배수가 될 때마다 주기적으로 삽입되며, 나머지 8개의 부호심볼들은 행 인덱스가 128( $=1024/8$ )의 배수가 될 때마다 주기적으로 삽입되므로, 결국 280개의 부호심볼들은 행 인덱스 128을 주기로 그 삽입 패턴이 동일하다고 할 수 있다. 따라서, 1024개의 행들에 280개의 부호심볼들을 삽입하는 것은, 매

128( $128(=1024/8)$ )개의 행들마다 35( $=280/8$ )개의 부호심볼들을 삽입하는 것과 동일하다고 축약 해석할 수 있다.

<123>      상기 삽입해야 할 부호심볼들의 개수 35를 2의 거듭제곱 형태로 표현하면,  $35=32+2+1$  혹은  $35=32+4-1$ 이다. 이 두 가지 경우에 대하여 주소 보상값을 각각 계산할 수 있을 것이나, 앞서 설명한  $R=24$ 인 경우에서와 마찬가지로 35를  $32+2+1$ 의 형태로 표현하면 주소값 보상기의 구현시 하나의 가산기가 더 필요하게 될 것이므로, 35를  $32+4-1$ 의 형태로 표현하는 경우에 대해서만 주소 보상값의 생성 원리를 설명하기로 한다.

<124>      35를  $32+4-1$ 로 표현한 경우, 1024개의 행들에 먼저 32개의 부호심볼들을 삽입하고 4개의 부호심볼들을 더 삽입한 후 1개의 부호심볼을 제거한다. 그러면 첫 번째 삽입시에  $d$ 는  $4(=128/32)$ 이고 두 번째 삽입시에  $d$ 는  $32(=128/4)$ 이며 제거시에  $d$ 는  $128(=128/1)$ 이다.

<125>       $d$ 가 4인 경우는 행렬의 마지막 열이 부호심볼을 전혀 전혀 가지고 있지 않은 상태에서 행간 거리가 4인 32개의 부호심볼들을 상기 마지막 열에 최초로 삽입하는 것이므로, 첫 번째 삽입되는 부호심볼의 행 인덱스는 0이고, 따라서 이 때의  $r^+$ 는 0이다.  $d$ 가 32인 경우는 행렬의 마지막 열이 이미 행간 거리가 4인 32개의 부호심볼들을 가지고 있는 상태에서 4개의 부호심볼들을 추가로 삽입하는 경우이므로 첫 번째 삽입되는 부호심볼은 BRO 연산의 특성상 이미 삽입된 32개의 부호심볼들 중 처음 두 개의 부호심볼들 사이의 중간에 삽입되므로, 따라서 이 때의  $r^+$ 는  $2(=(0+4)/2)$ 이다. 마지막으로  $d$ 가 128인 경우는 이미 삽입된 부호심볼들 중 가장 큰 행 인덱스를 가지는 1개의 부호심볼을 제거해야 하므로, 이 때 제거되는 부호심볼은 BRO 연산의 특성상 상기 두 번째로 삽입된 4개의 부호심볼들 중 가장 마지막 행의 부호심볼이 되고, 따라서  $r^-$ 는  $98(=2+3 \times 32)$ 이 된다.

<126> 결과적으로 앞서 언급한 <수학식 7>과 <수학식 9>를 이용하면, 부호화 패킷의 크기가 2328인 경우에 적용되는 주소 보상값 생성식은 하기의 <수학식 15>과 같다.

<127>

$$C(r_k) = C_4^+(r_k) + C_{32}^+(r_k) + C_{128}^-(r_k)$$

$$= \left[ \frac{r_k + 4 - (0+1)}{4} \right] + \left[ \frac{r_k + 32 - (2+1)}{32} \right] - \left[ \frac{r_k + 128 - (98+1)}{128} \right]$$

【수학식 15】

$$= \left[ \frac{r_k + 3}{4} \right] + \left[ \frac{r_k + 29}{32} \right] - \left[ \frac{r_k + 29}{128} \right]$$

<128> C.  $N_{EP}=3864$ 인 경우( $R=1816$ ) 주소 보상값의 생성

<129> 부호화 패킷의 크기가 3864인 경우, 하나의 서브블럭에 포함된 3864개의 부호심볼들은 2048( $=2^{11}$ )개의 행들을 가지는 행렬로 구성되고, 이 행렬의 마지막 열에서는 임시주소 생성시 함께 고려되었던 232( $=2048-1816$ )개의 부호심볼들이 제거되어야 한다. 232를 2의 거듭제곱 형태로 다시 표현하면  $2^7+2^6+2^5+2^3$ 이 되므로, 상기 232개의 부호심볼들을 128개와 64개와 32개와 8개로 나누어 고려하기로 한다.

<130> 즉, 232개의 부호심볼들 중 마지막 8개의 부호심볼들은 행 인덱스가 256( $=2048/8$ )의 배수가 될 때마다 주기적으로 제거되므로, 결국 232개의 부호심볼들은 256을 주기로 그 삽입 패턴이 동일하게 된다. 따라서, 2048개의 행들에서 232개의 부호심볼들을 제거하는 것은, 매 256개의 행들마다 29( $=232/8$ )의 부호심볼들을 제거하는 것과 동일하다고 추약 해석할 수 있다.

<131> 상기 제거해야할 부호심볼들의 개수 29를 2의 거듭제곱 형태로 표현하면,  $-29=-16-8-4-1$  혹은  $-29=-32+4-1$ 이다. 이들 두 가지 경우에 대하여 주소 보상값을 각각 계산할 수 있을 것이나,  $-29$ 를  $-16-8-4-1$ 의 형태로 표현하면 주소값 보상기의 구현시 몇 개의 가산기들이 더 필요하게 될 것이므로,  $-29$ 를  $-32+4-1$ 의 형태로 표현하는 경우에 대해서만 주소 보상값의 생성 원리를 설명하기로 한다.

<132> -29를 -32+4-1로 표현한 경우, 먼저 32개의 부호심볼들을 제거하고 다음으로 4개의 부호심볼들을 삽입한 후 마지막으로 1개의 부호심볼들을 제거한다. 그러면 첫 번째 제거시에  $d$ 는  $8(=256/32)$ 이고 삽입시에  $d$ 는  $64(=256/4)$ 이며 두 번째 제거시에  $d$ 는  $256(=256/1)$ 이다.

<133>  $d$ 가 8인 경우는 행렬의 마지막 열에 부호심볼들이 완전히 채워진 상태에서 행간 거리가 8인 32개의 부호심볼들을 상기 마지막 열에서 최초로 제거하는 것이므로, 첫 번째 제거되는 부호심볼의 행 인덱스는 BRO 연산의 특성상  $d-1$ 이며 따라서 이때의  $r^-$ 는  $7(=8-1)$ 이다. 또한  $d$ 가 64인 경우는 상기 제거된 32개의 부호심볼들 중 행간 거리가 64인 4개의 부호심볼들을 다시 복원하는 것인데, 이 때 최초로 복원되는 부호심볼은 BRO 연산의 특성상 상기 제거된 32개의 부호심볼들 중 가장 작은 행 인덱스를 가지는 부호심볼이므로 이 때의  $r^-$ 도 역시  $7(=8-1)$ 이 된다. 마지막으로  $d$ 가 256인 경우는 상기 복원된 4개의 부호심볼들 중 가장 큰 행 인덱스를 가지는 1개의 부호심볼을 다시 제거하는 것이므로, 이때의  $r^-$ 는  $199(=7+3 \times 64)$ 가 된다.

<134> 결과적으로 앞서 언급한 <수학식 7>과 <수학식 9>를 이용하면, 부호화 패킷의 크기가 3864인 경우에 적용되는 주소 보상값 생성식은 하기의 <수학식 16>과 같다.

<135>

$$\begin{aligned}
 C(r_k) &= C_8^-(r_k) + C_{64}^+(r_k) + C_{256}^-(r_k) \\
 &= -\left[ \frac{r_k + 8 - (7+1)}{8} \right] + \left[ \frac{r_k + 64 - (7+1)}{64} \right] - \left[ \frac{r_k + 256 - (199+1)}{256} \right] \\
 &= -\left[ \frac{r_k}{8} \right] + \left[ \frac{r_k + 56}{64} \right] - \left[ \frac{r_k + 56}{256} \right]
 \end{aligned}$$

【수학식 16】

<136> 이상에서 상세히 설명한 바와 같이 1xEV-DV 표준을 지원하는 단말기에서 순방향 패킷 데이터 트래픽에 대한 서브블럭 디인터리빙은, 임시주소 생성기(322)가 <수학식 5>와 <수학식 6>에 따라 임시주소를 생성하면, 주소 보상기(324)는 <수학식 14>, <수학식 15>, <수학식 16>

중 하나를 이용하여 주소 보상값을 생성하고, 가산기(326)는 <수학식 4>를 계산하여 최종 읽기주소를 생성함으로써 수행된다.

<137> 즉, <수학식 6>과 부호화 패킷의 크기에 따라 선택된 <수학식 14>, <수학식 15>, <수학식 16>를 <수학식 4>에 대입하여 정리하면, 하기의 <수학식 17>, <수학식 18>, <수학식 19>을 얻을 수 있다. <수학식 17>은 부호화 패킷의 크기가  $408(m=7)$ ,  $792(m=8)$ ,  $1560(m=9)$ ,  $3096(m=10)$ 인 경우, <수학식 18>은 부호화 패킷의 크기가 2328인 경우, <수학식 19>는 부호화 패킷의 크기가 3864인 경우 각각에 대한 입력버퍼(310)의 읽기주소 생성식이다.

<138>

$$A_k = IA_k + C(r_k)$$

$$= (J-1) \cdot BRO_m(k \bmod 2^m) + \lceil k/2^m \rceil$$

$$+ \left[ \frac{BRO_m(k \bmod 2^m) + 2^{m-5} - 1}{2^{m-5}} \right] - \left[ \frac{BRO_m(k \bmod 2^m) + 2^{m-5} - 1}{2^{m-3}} \right]$$

【수학식 17】

<139>

$$A_k = IA_k + C(r_k)$$

$$= 2 \cdot BRO_{10}(k \bmod 2^{10}) + \left\lceil \frac{k}{2^{10}} \right\rceil$$

$$+ \left[ \frac{BRO_{10}(k \bmod 2^{10}) + 3}{4} \right]$$

$$+ \left[ \frac{BRO_{10}(k \bmod 2^{10}) + 29}{32} \right] - \left[ \frac{BRO_{10}(k \bmod 2^{10}) + 29}{128} \right]$$

【수학식 18】

<140>

$$A_k = IA_k + C(r_k)$$

$$= 2 \cdot BRO_{11}(k \bmod 2^{11}) + \left\lceil \frac{k}{2^{11}} \right\rceil$$

$$- \left[ \frac{BRO_{11}(k \bmod 2^{11})}{8} \right]$$

$$+ \left[ \frac{BRO_{11}(k \bmod 2^{11}) + 56}{64} \right] - \left[ \frac{BRO_{11}(k \bmod 2^{11}) + 56}{256} \right]$$

【수학식 19】

<141> 도 10은 본 발명에 따라 입력버퍼(310)를 위한 읽기주소를 생성하는 디인터리빙 주소 발생기(320)에서 부호화 패킷의 크기가  $408(m=7)$ ,  $792(m=8)$ ,  $1560(m=9)$ ,  $3096(m=10)$ 이고 J는 4인 경우, 읽기주소를 발생하기 위한 상세 구성을 나타낸 것으로서, <수학식 17>을 하드웨어 장치로 구현한 것이다.

- <142> 도시한 바와 같이, 채널 복호기(320)로부터 입력받고자 하는 부호심볼 인덱스  $k$ 가 임시 주소 생성기(322a)로 입력되면, 제산기(400)는 상기 부호심볼 인덱스  $k$ 를  $2^m$ 으로 나누어 가산기(406)로 제공하고, BRO 연산기(402)는 상기 부호심볼 인덱스  $k$ 를  $2^m$ 으로 나눈 나머지를  $m$ 비트 단위로 비트 역전환하여 상기 부호심볼 인덱스  $k$ 에 대한 행 인덱스  $r_k$ 를 계산한다. 승산기(404)는 상기 행 인덱스  $r_k$ 에  $3(=J-1)$ 을 곱하여 상기 가산기(406)로 제공한다. 그러면 가산기(406)는 상기 제산기(400)의 출력  $k/2^m$ 에 상기 승산기(404)의 출력을 더하여 임시주소  $IA_k$ 을 계산한다.
- <143> 상기 행 인덱스  $r_k$ 가 주소 보상기(324a)로 입력되면, 가산기(410)는 상기 행 인덱스  $r_k$ 에  $2^{m-5}-1$ 을 더하여 제산기들(412, 414)로 제공한다. 제산기(412)는 상기 가산기(410)의 출력  $(r_k + 2^{m-5}-1)$ 을  $2^{m-5}$ 로 나누며, 제산기(414)는 상기 가산기(410)의 출력  $(r_k + 2^{m-5}-1)$ 을  $2^{m-3}$ 으로 나눈다. 가산기(416)는 상기 제산기(412)의 출력에서 상기 제산기(414)의 출력을 빼서 주소 보상값  $C(r_k)$ 를 계산한다. 가산기(326)는 상기 임시주소  $IA_k$ 에 상기 주소 보상값  $C(r_k)$ 를 더하여 최종적으로 읽기주소  $A_k$ 를 계산한다.
- <144> 도 11은 본 발명에 따라 입력버퍼(310)를 위한 읽기주소를 생성하는 디인터리빙 주소 발생기(320)에서 부호화 패킷의 크기가 2328( $m=10$ ,  $J=3$ )인 경우, 읽기주소를 발생하기 위한 상세 구성을 나타낸 것으로서, <수학식 18>을 하드웨어 장치로 구현한 것이다.
- <145> 도시한 바와 같이, 채널 복호기(320)로부터 입력받고자 하는 부호심볼 인덱스  $k$ 가 임시 주소 생성기(322b)로 입력되면, 제산기(420)는 상기 부호심볼 인덱스  $k$ 를  $2^m$ 으로 나누어 가산기(426)로 제공하고, BRO 연산기(422)는 상기 부호심볼 인덱스  $k$ 를  $2^m$ 으로 나눈 나머지를  $m$ 비트 단위로 비트 역전환하여 상기 부호심볼 인덱스  $k$ 에 대한 행 인덱스  $r_k$ 를 계산한다. 승산기

(424)는 상기 행 인덱스  $r_k$ 에  $3(=J-1)$ 을 곱하여 상기 가산기(426)로 제공한다. 그러면 가산기(426)는 상기 제산기(420)의 출력  $k/2^m$ 에 상기 승산기(424)의 출력을 더하여 임시주소  $IA_k$ 을 계산한다.

<146>      상기 행 인덱스  $r_k$ 가 주소 보상기(324b)로 입력되면, 가산기(430)는 상기 행 인덱스  $r_k$ 에 3을 더하고, 제산기(434)는 상기 가산기(430)의 출력  $(r_k+3)$ 을 4로 나누어 출력한다. 또한 가산기(432)는 상기 행 인덱스  $r_k$ 에 29를 더하고, 제산기(436)는 상기 가산기(432)의 출력  $(r_k+29)$ 을 32로 나누어 출력하며, 제산기(438)는 상기 가산기(432)의 출력  $(r_k+29)$ 을 128로 나누어 출력한다. 가산기(440)는 상기 제산기(434)의 출력에 상기 제산기(436)의 출력을 더하고 상기 제산기(438)의 출력을 빼서 주소 보상값  $C(r_k)$ 를 계산한다. 가산기(326)는 상기 임시주소  $IA_k$ 에 상기 주소 보상값  $C(r_k)$ 를 더하여 최종적으로 읽기주소  $A_k$ 를 계산한다.

<147>      도 12는 본 발명에 따라 입력버퍼(310)를 위한 읽기주소를 생성하는 디인터리빙 주소 발생기(320)에서 부호화 패킷의 크기가 3864( $m=11$ ,  $J=2$ )인 경우, 읽기주소를 발생하기 위한 상세 구성을 나타낸 것으로서, <수학식 19>을 하드웨어 장치로 구현한 것이다.

<148>      도시한 바와 같이, 채널 복호기(320)로부터 입력받고자 하는 부호심볼 인덱스  $k$ 가 임시주소 생성기(322c)로 입력되면, 제산기(450)는 상기 부호심볼 인덱스  $k$ 를  $2^m$ 으로 나누어 가산기(456)로 제공하고, BRO 연산기(452)는 상기 부호심볼 인덱스  $k$ 를  $2^m$ 으로 나눈 나머지를  $m$ 비트 단위로 비트 역전환하여 상기 부호심볼 인덱스  $k$ 에 대한 행 인덱스  $r_k$ 를 계산한다. 승산기(454)는 상기 행 인덱스  $r_k$ 에  $3(=J-1)$ 을 곱하여 상기 가산기(456)로 제공한다. 그러면 가산기(456)는 상기 제산기(450)의 출력  $k/2^m$ 에 상기 승산기(454)의 출력을 더하여 임시주소  $IA_k$ 을 계산한다.

<149>      상기 행 인덱스  $r_k$ 가 주소 보상기(324c)로 입력되면, 제산기(462)는 상기 행 인덱스  $r_k$ 를 8로 나누고, 가산기(460)는 상기 행 인덱스  $r_k$ 에 56을 더하여 출력한다. 제산기(464)는 상기 가산기(460)의 출력 ( $r_k+56$ )을 64로 나누고, 제산기(466)는 상기 가산기(460)의 출력 ( $r_k+56$ )을 256으로 나누어 출력한다. 가산기(468)는 상기 제산기(464)의 출력에서 상기 제산기(462)의 출력과 상기 제산기(466)의 출력을 빼서 주소 보상값  $C(r_k)$ 를 계산한다. 가산기(326)는 상기 임시주소  $IA_k$ 에 상기 주소 보상값  $C(r_k)$ 를 더하여 최종적으로 읽기주소  $A_k$ 를 계산한다.

<150>      한편 본 발명의 상세한 설명에서는 구체적인 실시예에 관해 설명하였으나, 본 발명의 범위에서 벗어나지 않는 한도 내에서 여러 가지 변형이 가능함은 물론이다. 그러므로 본 발명의 범위는 설명된 실시예에 국한되지 않으며, 후술되는 특허청구의 범위뿐만 아니라 이 특허청구의 범위와 균등한 것들에 의해 정해져야 한다.

### 【발명의 효과】

<151>      이상에서 상세히 설명한 바와 같이 동작하는 본 발명에 있어서, 개시되는 발명중 대표적인 것에 의하여 얻어지는 효과를 간단히 설명하면 다음과 같다.

<152>      본 발명은, 1xEV-DV 표준에 규정된 디인터리빙 규칙을 사용하는 통신 시스템에서 인터리빙된 데이터의 열을 고속으로 디인터리빙하여 원래의 부호화 패킷을 복구할 수 있다. 특히 본 발명은 수신된 부호심볼들의 서브블럭들을 메모리에 저장하고, 디인터리빙 규칙에 따라 생성된 읽기주소에 따라 부호심볼들을 읽어내어 복호기로 제공함으로써 필요한 메모리의 개수를 최소화하고, 읽기주소를 계산하는데 필요한 하드웨어 소자의 개수를 최소화하여 수신기의 소형화,



저전력화를 가능하게 한다. 또한 본 발명은 각각의 서브블럭별로 디인터리빙을 수행하기 때문에 복호기에서 복수의 서브블럭들을 동시에 복호함으로써 복호속도를 더욱 향상시킬 수 있으므로 고속 데이터 통신을 가능하게 한다는 효과가 있다.

**【특허청구범위】****【청구항 1】**

인터리빙된 순서대로 버퍼에 저장된 한 서브블럭의 부호심볼들을 디인터리빙된 순서대로 읽어내어 채널 복호기로 입력하는 방법에 있어서, 여기서 상기 부호심볼들은 상기 서브블럭의 크기에 따라 미리 정해지는 개수의 행과 열을 가지는 행렬의 형태로 저장되어 있고,

상기 행렬의 마지막 열을 고려하지 않고, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 임시주소를 생성하는 과정과,

상기 행렬의 마지막 열을 고려하여 상기 임시주소를 보상하기 위한 주소 보상값을 계산하는 과정과,

상기 임시주소에 상기 주소 보상값을 더하여, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 읽기주소를 생성하는 과정을 포함하는 것을 특징으로 하는 상기 방법.

**【청구항 2】**

제 1 항에 있어서, 상기 임시주소를 생성하는 과정은,

상기 마지막 열이 절반 이상의 부호심볼들을 가지지 않는 경우 상기 마지막 열을 생략하여 상기 임시주소를 생성하고, 상기 마지막 열이 절반 이상의 부호심볼들을 가지는 경우 상기 마지막 열을 포함하여 상기 임시주소를 생성하는 것을 특징으로 하는 상기 방법.

## 【청구항 3】

제 2 항에 있어서, 상기 주소 보상값을 계산하는 과정은,

상기 주소 보상값을 0으로 설정하고, 상기 마지막 열이 절반 이상의 부호심볼들을 가지지 않는 경우 상기 마지막 열에서 부호심볼이 나타날 때마다 상기 주소 보상값을 1씩 증가시키며, 상기 마지막 열이 절반 이상의 부호심볼들을 가지는 경우 상기 마지막 열에서 부호심볼을 생략할 때마다 상기 주소 보상값을 1씩 감소시키는 것을 특징으로 하는 상기 방법.

## 【청구항 4】

제 1 항에 있어서, 상기 서브블럭의 크기가 408인 경우, 상기 읽기주소는 하기의 <수학식 20>에 의해 생성되는 것을 특징으로 하는 상기 방법.

$$\begin{aligned} A_k = & 3 \cdot BRO_7(k \bmod 128) + [k/128] \\ & + \left[ \frac{BRO_7(k \bmod 128) + 3}{4} \right] - \left[ \frac{BRO_7(k \bmod 128) + 3}{16} \right] \end{aligned}$$

【수학식 20】

여기서  $A_k$ 는 상기 읽기주소이고  $k$ 는 상기 채널 복호기에 의해 요구된 부호심볼의 인덱스이며, BRO는 비트 역전환(Bit Reversal Ordering) 연산을 의미하고 mod는 모듈로 연산을 의미하고  $[\cdot]$ 는 입력을 초과하지 않는 최대의 정수를 의미함.

## 【청구항 5】

제 1 항에 있어서, 상기 서브블럭의 크기가 792인 경우, 상기 읽기주소는 하기의 <수학식 21>에 의해 생성되는 것을 특징으로 하는 상기 방법.

$$\begin{aligned} A_k &= 3 \cdot BRO_8(k \bmod 256) + [k/256] \\ &+ \left[ \frac{BRO_8(k \bmod 256) + 7}{8} \right] - \left[ \frac{BRO_8(k \bmod 256) + 7}{32} \right] \end{aligned}$$

【수학식 21】

여기서  $A_k$ 는 상기 읽기주소이고  $k$ 는 상기 채널 복호기에 의해 요구된 부호심볼의 인덱스이며,  $BRO$ 는 비트 역전환 연산을 의미하고  $\bmod$ 는 모듈로 연산을 의미하고  $[ \cdot ]$ 는 입력을 초과하지 않는 최대의 정수를 의미함.

#### 【청구항 6】

제 1 항에 있어서, 상기 서브블럭의 크기가 1560인 경우, 상기 읽기주소는 하기의 <수학식 22>에 의해 생성되는 것을 특징으로 하는 상기 방법.

$$\begin{aligned} A_k &= 3 \cdot BRO_9(k \bmod 512) + [k/512] \\ &+ \left[ \frac{BRO_9(k \bmod 512) + 15}{16} \right] - \left[ \frac{BRO_9(k \bmod 512) + 15}{64} \right] \end{aligned}$$

【수학식 22】

여기서  $A_k$ 는 상기 읽기주소이고  $k$ 는 상기 채널 복호기에 의해 요구된 부호심볼의 인덱스이며,  $BRO$ 는 비트 역전환 연산을 의미하고  $\bmod$ 는 모듈로 연산을 의미하고  $[ \cdot ]$ 는 입력을 초과하지 않는 최대의 정수를 의미함.

#### 【청구항 7】

제 1 항에 있어서, 상기 서브블럭의 크기가 3096인 경우, 상기 읽기주소는 하기의 <수학식 23>에 의해 생성되는 것을 특징으로 하는 상기 방법.

$$\begin{aligned} A_k &= 3 \cdot BRO_{10}(k \bmod 1024) + [k/1024] \\ &+ \left[ \frac{BRO_{10}(k \bmod 1024) + 31}{32} \right] - \left[ \frac{BRO_{10}(k \bmod 1024) + 31}{128} \right] \end{aligned}$$

【수학식 23】

여기서  $A_k$ 는 상기 읽기주소이고  $k$ 는 상기 채널 복호기에 의해 요구된 부호심볼의 인덱스이며, BRO는 비트 역전환 연산을 의미하고 mod는 모듈로 연산을 의미하고  $[\cdot]$ 는 입력을 초과하지 않는 최대의 정수를 의미함.

#### 【청구항 8】

제 1 항에 있어서, 상기 서브블럭의 크기가 2328인 경우, 상기 읽기주소는 하기의 <수학식 24>에 의해 생성되는 것을 특징으로 하는 상기 방법.

$$\begin{aligned} A_k = & 2 \cdot BRO_{10}(k \bmod 2^{10}) + \left[ \frac{k}{2^{10}} \right] + \left[ \frac{BRO_{10}(k \bmod 2^{10}) + 3}{4} \right] \\ & + \left[ \frac{BRO_{10}(k \bmod 2^{10}) + 29}{32} \right] - \left[ \frac{BRO_{10}(k \bmod 2^{10}) + 29}{128} \right] \end{aligned}$$

【수학식 24】

여기서  $A_k$ 는 상기 읽기주소이고  $k$ 는 상기 채널 복호기에 의해 요구된 부호심볼의 인덱스이며, BRO는 비트 역전환 연산을 의미하고 mod는 모듈로 연산을 의미하고  $[\cdot]$ 는 입력을 초과하지 않는 최대의 정수를 의미함.

#### 【청구항 9】

제 1 항에 있어서, 상기 서브블럭의 크기가 3864인 경우, 상기 읽기주소는 하기의 <수학식 25>에 의해 생성되는 것을 특징으로 하는 상기 방법.

$$\begin{aligned} A_k = & 2 \cdot BRO_{11}(k \bmod 2^{11}) + \left[ \frac{k}{2^{11}} \right] - \left[ \frac{BRO_{11}(k \bmod 2^{11})}{8} \right] \\ & + \left[ \frac{BRO_{11}(k \bmod 2^{11}) + 56}{64} \right] - \left[ \frac{BRO_{11}(k \bmod 2^{11}) + 56}{256} \right] \end{aligned}$$

【수학식 25】

여기서  $A_k$ 는 상기 읽기주소이고  $k$ 는 상기 채널 복호기에 의해 요구된 부호심볼의 인덱스이며, BRO는 비트 역전환 연산을 의미하고 mod는 모듈로 연산을 의미하고  $[\cdot]$ 는 입력을 초과하

지 않는 최대의 정수를 의미함.

#### 【청구항 10】

인터리빙된 순서대로 버퍼에 저장된 한 서브블럭의 부호심볼들을 디인터리빙된 순서대로 읽어내어 채널 복호기로 입력하는 장치에 있어서, 여기서 상기 부호심볼들은 상기 서브블럭의 크기에 따라 미리 정해지는 개수의 행과 열을 가지는 행렬의 형태로 저장되어 있고,

상기 행렬의 마지막 열을 고려하지 않고, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 임시주소를 생성하는 임시주소 생성기와,

상기 행렬의 마지막 열을 고려하여 상기 임시주소를 보상하기 위한 주소 보상값을 계산하는 주소 보상기와,

상기 임시주소에 상기 주소 보상값을 더하여, 상기 채널 복호기에 의해 요구된 부호심볼을 상기 버퍼로부터 읽어내기 위한 읽기주소를 생성하는 가산기를 포함하는 것을 특징으로 하는 상기 장치.

#### 【청구항 11】

제 10 항에 있어서, 상기 서브블럭의 크기는  $2^m \times (J-1) + R$ 이고 여기서 상기  $m$ 과  $J$ 와  $R$ 은 상기 서브블럭의 크기에 따라 미리 정해질 때, 상기 임시주소 생성기는,

상기 채널 복호기에 의해 요구된 부호심볼의 인덱스를  $2^m$ 으로 나누어 정수 형태로 출력하는 제1 계산기와,

상기 부호심볼 인덱스를  $2^m$ 으로 나눈 나머지를 비트 역전환하는 BRO 연산기와,

상기 BRO 연산기로부터의 출력에 (J-1)를 곱하는 승산기와,

상기 제1 제산기로부터의 출력에 상기 승산기로부터의 출력을 더하여 상기 임시주소를 계산하는 제1 가산기로 구성되는 것을 특징으로 하는 상기 장치.

【청구항 12】

제 11 항에 있어서, 상기 서브블럭의 크기가 408인 경우, 상기 주소 보상기는,

상기 BRO 연산기로부터의 출력에 3을 더하는 제2 가산기와,

상기 제2 가산기로부터의 출력을 4로 나누어 정수 형태로 출력하는 제2 제산기와,

상기 제2 가산기로부터의 출력을 16으로 나누어 정수 형태로 출력하는 제3 제산기와,

상기 제2 제산기로부터의 출력에서 상기 제3 제산기로부터의 출력을 빼서 상기 주소 보상값을 계산하는 가산기로 구성되는 것을 특징으로 하는 상기 장치.

【청구항 13】

제 11 항에 있어서, 상기 서브블럭의 크기가 792인 경우, 상기 주소 보상기는,

상기 BRO 연산기로부터의 출력에 7을 더하는 제2 가산기와,

상기 제2 가산기로부터의 출력을 8로 나누어 정수 형태로 출력하는 제2 제산기와,

상기 제2 가산기로부터의 출력을 32로 나누어 정수 형태로 출력하는 제3 제산기와,

상기 제2 제산기로부터의 출력에서 상기 제3 제산기로부터의 출력을 빼서 상기 주소 보상값을 계산하는 가산기로 구성되는 것을 특징으로 하는 상기 장치.

## 【청구항 14】

제 11 항에 있어서, 상기 서브블럭의 크기가 1560인 경우, 상기 주소 보상기는,  
상기 BRO 연산기로부터의 출력에 15를 더하는 제2 가산기와,  
상기 제2 가산기로부터의 출력을 16으로 나누어 정수 형태로 출력하는 제2 제산기와,  
상기 제2 가산기로부터의 출력을 64로 나누어 정수 형태로 출력하는 제3 제산기와,  
상기 제2 제산기로부터의 출력에서 상기 제3 제산기로부터의 출력을 빼서 상기 주소 보상값을 계산하는 가산기로 구성되는 것을 특징으로 하는 상기 장치.

## 【청구항 15】

제 11 항에 있어서, 상기 서브블럭의 크기가 3096인 경우, 상기 주소 보상기는,  
상기 BRO 연산기로부터의 출력에 31을 더하는 제2 가산기와,  
상기 제2 가산기로부터의 출력을 32로 나누어 정수 형태로 출력하는 제2 제산기와,  
상기 제2 가산기로부터의 출력을 128로 나누어 정수 형태로 출력하는 제3 제산기와,  
상기 제2 제산기로부터의 출력에서 상기 제3 제산기로부터의 출력을 빼서 상기 주소 보상값을 계산하는 가산기로 구성되는 것을 특징으로 하는 상기 장치.

## 【청구항 16】

제 11 항에 있어서, 상기 서브블럭의 크기가 2328인 경우, 상기 주소 보상기는,  
상기 BRO 연산기로부터의 출력에 3을 더하는 제2 가산기와,  
상기 제2 가산기로부터의 출력을 4로 나누는 제2 제산기와,

상기 BRO 연산기로부터의 출력에 29를 더하는 제3 가산기와,

상기 제3 가산기로부터의 출력을 32로 나누는 제3 제산기와,

상기 제3 가산기로부터의 출력을 128로 나누는 제4 제산기와,

상기 제2 제산기로부터의 출력에 상기 제3 제산기로부터의 출력을 더하고 상기 제4 제산기로부터의 출력을 빼서 상기 주소 보상값을 계산하는 제4 가산기로 구성되는 것을 특징으로 하는 상기 장치.

【청구항 17】

제 11 항에 있어서, 상기 서브블럭의 크기가 3864인 경우, 상기 주소 보상기는,

상기 BRO 연산기로부터의 출력을 8로 나누는 제2 제산기와,

상기 BRO 연산기로부터의 출력에 56을 더하는 제2 가산기와,

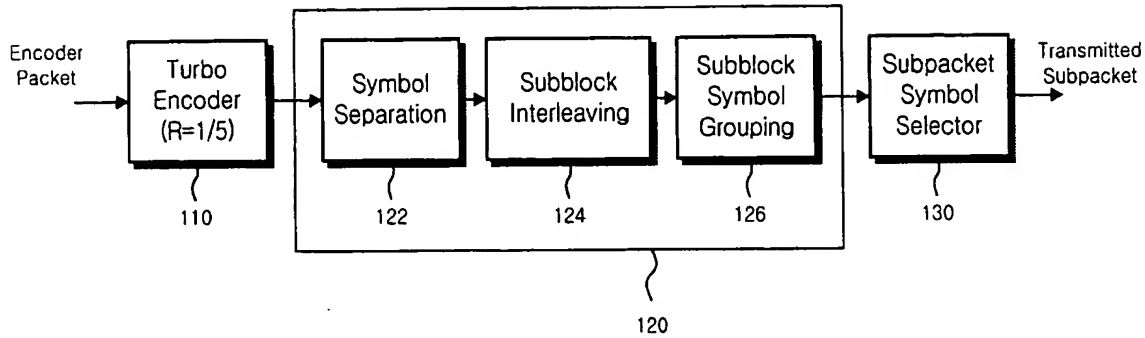
상기 제2 가산기로부터의 출력을 64로 나누는 제3 제산기와,

상기 제2 가산기로부터의 출력을 256으로 나누는 제4 제산기와,

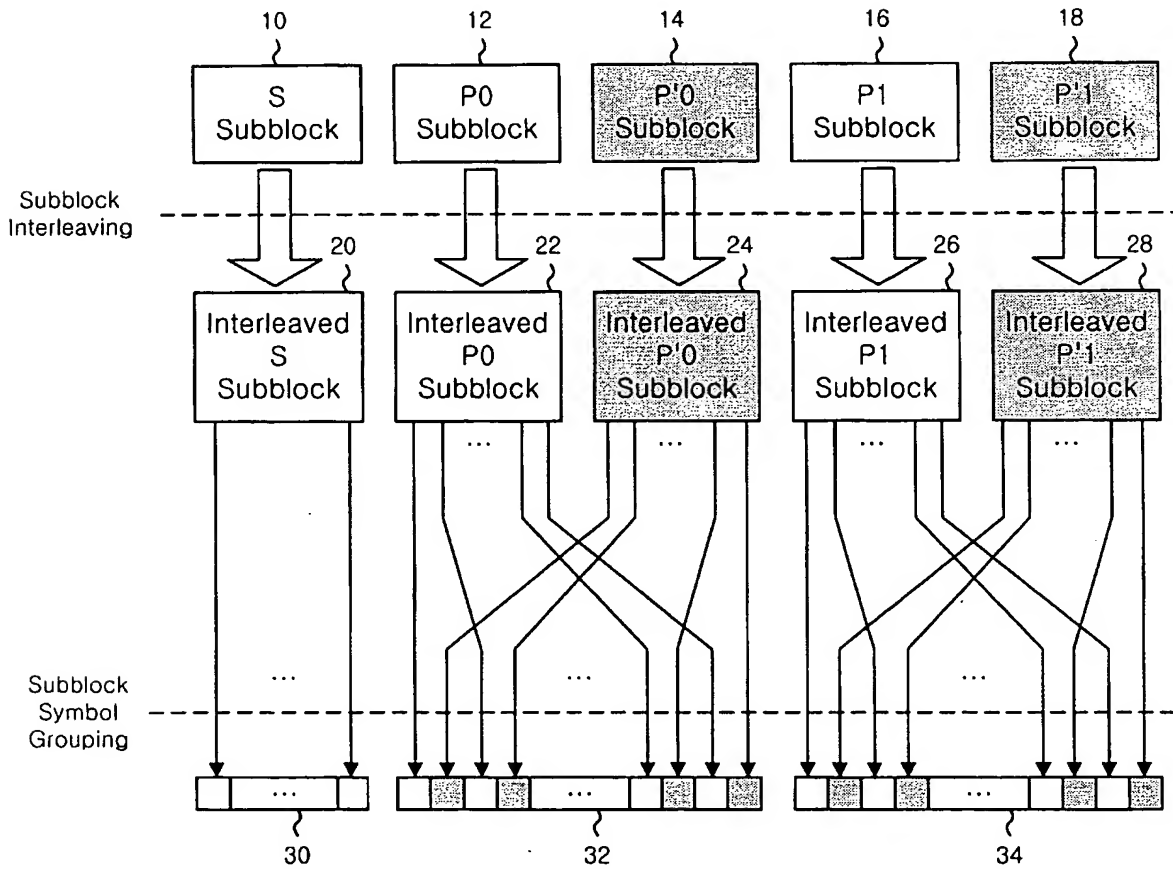
상기 제3 제산기로부터의 출력에서 상기 제2 제산기로부터의 출력과 상기 제4 제산기로부터의 출력을 빼서 상기 주소 보상값을 계산하는 제3 가산기로 구성되는 것을 특징으로 하는 상기 장치.

## 【도면】

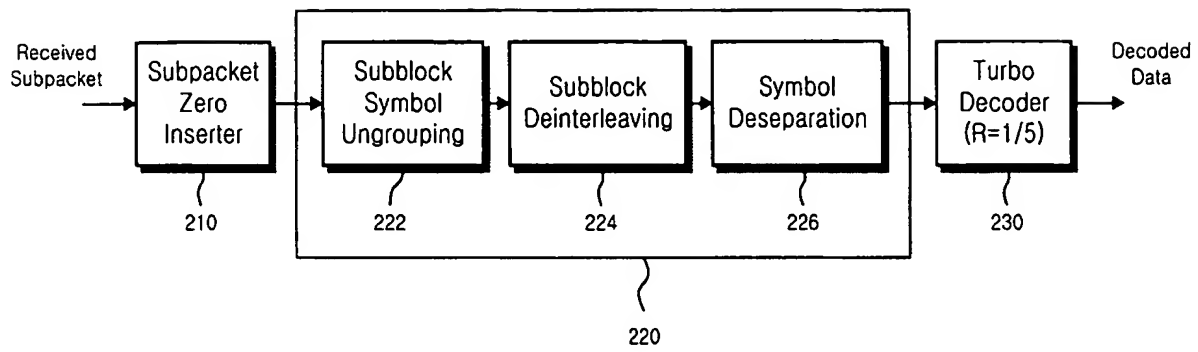
【도 1】



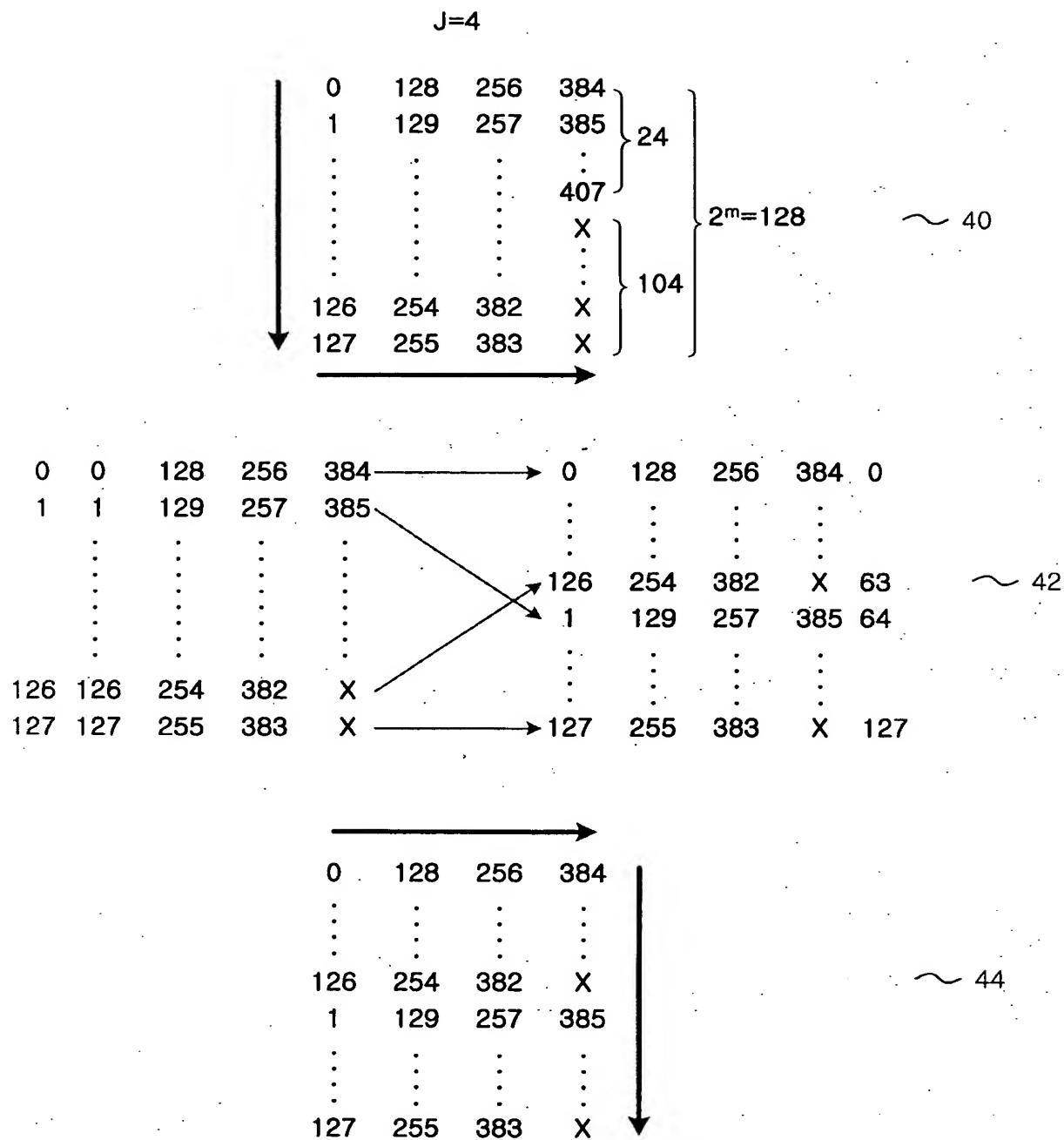
【도 2】



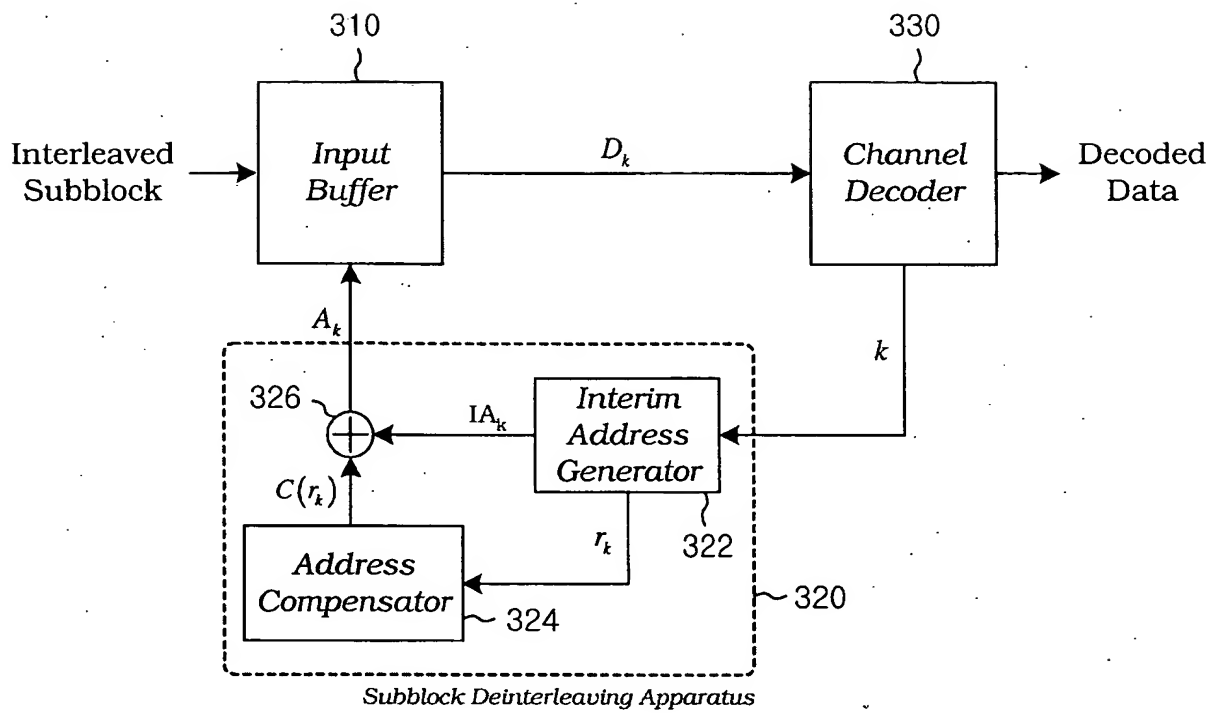
【도 3】



【도 4】



【도 5】



【도 6】

Example) m=4, J=2, and R=4

Row Index	Before Row BRO	After Row BRO
0	0 16	0 16
1	1 17	8
2	2 18	4
3	3 19	12
4	4	2 18
5	5	10
6	6	6
7	7	14
8	8	1 17
9	9	9
10	10	5
11	11	13
12	12	3 19
13	13	11
14	14	7
15	15	15

【표 7】

C.S. Index	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
: $k$	16				18				17				19			
Row Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
: $r_k$																
Interim Addr.	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
: $l_k$	[1]				[5]				[9]					[13]		
Comp. Factor	0	1	1	1	1	1	2	2	2	2	3	3	3	3	4	4
: $C(r_k)$																
Read Addr.	[0]	[2]	[3]	[4]	[5]	[7]	[8]	[9]	[10]	[12]	[13]	[14]	[15]	[17]	[18]	[19]
: $A_k$	[1]				[6]				[11]					[16]		

【도 8】

Example) m=4, J=2, and R=12

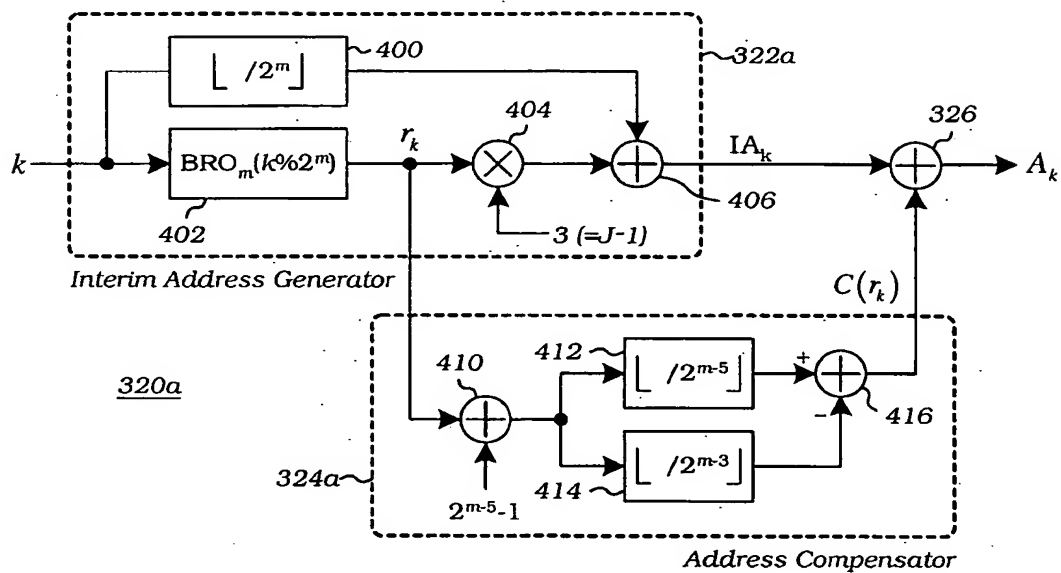
Row Index	Before Row BRO	After Row BRO
0	0 16	0 16
1	1 17	8 24
2	2 18	4 20
3	3 19	12 <span style="border: 1px solid black;">28</span>
4	4 20	2 18
5	5 21	10 26
6	6 22	6 22
7	7 23	14 <span style="border: 1px solid black;">30</span>
8	8 24	1 17
9	9 25	9 25
10	10 26	5 21
11	11 27	13 <span style="border: 1px solid black;">29</span>
12	12 <span style="border: 1px solid black;">28</span>	3 19
13	13 <span style="border: 1px solid black;">29</span>	11 27
14	14 <span style="border: 1px solid black;">30</span>	7 23
15	15 <span style="border: 1px solid black;">31</span>	15 <span style="border: 1px solid black;">31</span>

【 9】

C.S. Index	0	8	4	12	2	10	6	14	1	9	5	13	3	11	7	15
:k	16	24	20	28	18	26	22	30	17	25	21	29	19	27	23	31
Row Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
:r <sub>k</sub>																
Interim Addr.	[0]	[2]	[4]	[6]	[8]	[10]	[12]	[14]	[16]	[18]	[20]	[22]	[24]	[26]	[28]	[30]
:lA <sub>k</sub>	[1]	[3]	[5]		[9]	[11]	[13]		[17]	[19]	[21]		[25]	[27]	[29]	
Comp. Factor	0	0	0	0	-1	-1	-1	-1	-2	-2	-2	-2	-3	-3	-3	-3
:C(r <sub>k</sub> )																
Read Addr.	[0]	[2]	[4]	[6]	[7]	[9]	[11]	[13]	[14]	[16]	[18]	[20]	[21]	[23]	[25]	[27]
:A <sub>k</sub>	[1]	[3]	[5]		[8]	[10]	[12]		[15]	[17]	[19]		[22]	[24]	[26]	

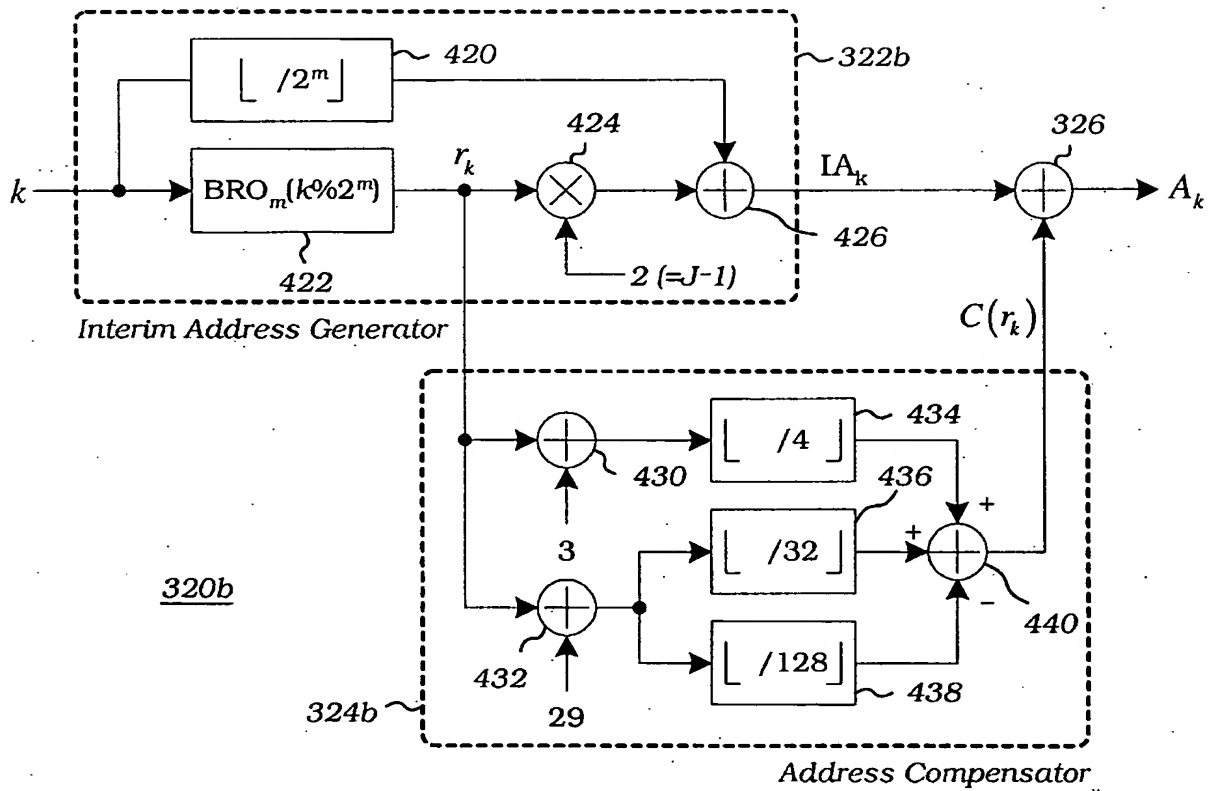
## 【도 10】

For the case of  $(N_{EP}=408, m=7, J=4)$ ,  $(N_{EP}=792, m=8, J=4)$ ,  
 $(N_{EP}=1560, m=9, J=4)$ , or  $(N_{EP}=3096, m=10, J=4)$



【도 11】

For the case of  $(N_{EP}=2328, m=10, J=3)$



【도 12】

For the case of ( $N_{EP}=3864$ ,  $m=11$ ,  $J=2$ )

